Welcome to

Sample Wrench

An Audio Wave Processor

dissidents

http://www.dissidents.com

Preface

You've heard of word processing? Welcome to Wave Processing. Sample Wrench has been designed to be a powerful and easy to use professional audio sample editor from the outset. Audio waves are stored and manipulated in either 16 bit format, yielding CD quality, or advanced 32 bit floating point format with 24 bits of precision and a dynamic range in excess of 190 dB, for the most demanding applications. Wrench, like all dissidents products, is the direct result of our own need and desire for such a system. The people who created Sample Wrench are trained musicians, as well as being professional programmers and engineers. Whatever your background, we hope that Sample Wrench fills your needs and exceeds your expectations.

We would like to thank the technical staffs of the various sampler manufacturers for their most important input. We would also like to thank nature for providing the bananas, strawberries, and kiwi fruit, and George Washington Carver for all the neat things he did with peanuts.

This manual is dedicated to DIYers everywhere.

Serial Numbers and Updates

Unlike earlier versions of this software, a serial number is not required to activate this program. Updates are released periodically, with new features added in accordance to demand, so if there's something you want, let us know! Info on new products, promos, freebies, and technical support are available on our web site: http://www.dissidents.com.

System Requirements

Computer:

Sample Wrench generally places modest demands on your computer. Almost any Windows PC produced since the late 1990s will run Sample Wrench. At a minimum, a Pentium II class CPU is required.

Operating System:

Windows 95, 98, etc. is required. Newer versions (e.g., 8 and 10) may require downloading drivers for the on-line help system.

Memory and Drives:

A minimum of 64 Meg RAM is required. Generally, the more RAM you have the faster Wrench will run, especially if you edit large sound samples. The installation requires approximately 6 Meg of hard drive space.

Display Monitor:

Any compatible monitor should work fine.

Audio Playback:

A Windows compatible sound card or internal audio is strongly recommended (with a MIDI interface for sampler transfer, if desired). Since Wrench allows you to preview sounds, some form of sound output is desired. This can range from the internal speakers found in some display monitors, to a direct connection to a Hi-Fi system. For final critical listening with MIDI based samplers, sounds should be downloaded to the sampler and then played back through an appropriate studio monitoring system.

A Note on Multi-tasking:

Wrench will happily multi-task with other programs. It is important to remember that Wrench will need to access system resources from time to time (such as MIDI ports or audio playback hardware). Where possible, Wrench tries to share system resources, but this is not always possible (or even desirable). If Wrench needs exclusive access to a resource it will only do so when necessary, freeing up the system for other applications when possible. One important consideration involves large sound sample transfers via MIDI system exclusive (SYS-EX) messages. You should not attempt to use other MIDI music programs while you are sending or receiving MIDI data as Wrench requires lone access to the MIDI port during this process. If this were not the case, a simple MIDI Note On message sent by another music program could abort the system exclusive message (as detailed in the MIDI Spec). The result would be chaotic, to say the least.

Installing Sample Wrench

To install Wrench onto your hard disk, simply run the program that you downloaded (it's a self-installer). By default, it will create a directory on your hard disk called "dissidents" which contains the application directory "Wrench2496", and copy over all of the files required. Inside "Wrench" will be sub-directories called "Program" (where Wrench itself is), "Presets" (for example effects presets), "Scripts" (example macros) and "Sounds" (for example and tutorial sounds). Usage of the installer is quite straightforward. An Uninstaller is also provided.

Sample Wrench XE does not use any form of intrusive copy protection, ads or similar devices. If you like the program we suggest you consider a modest donation to help with further updates and support.

Using the Manual

The entire manual is available on-line. It may be accessed via the Help menu, or through the context-sensitive Help buttons found in many dialog boxes. The tutorial section is reproduced in the following pages for your convenience. Here is a description of each section:

Tutorial & Overview

This is a practical, hands-on look at Sample Wrench and what it can do. It touches on many of the important features but is by no means an exhaustive workout. It should give you a good feel for the flow of the program. Even if you're an old-timer with music and computers, we strongly suggest that you walk through this.

Fundamentals

This section details the modes of operation and general settings. It introduces the topics and gives how-to explanations. Details on recording, playback, file formats, markers and loops may be found here.

Functions

This gives details on the items found under the Functions menu, including EQ, level control, noise reduction, sample rate alterations, keymapping and looping aids.

Effects

This gives detail on the items found under the Effects menu, including AM, FM, reverb, chorus, impulse modeling, pitch shifting, time compression, and spectral warping.

General Reference

Information on Wrench's startup characteristics, utilities and shortcut keys may be found here.

Keyboard Samplers

This section gives details on transferring sounds to or from a MIDI or SMDI based sampler.

Frequently Asked Questions

This is a listing of commonly asked questions and answers concerning Sample Wrench and digital audio.

Wrench-specific Enable Scripting

This section shows the ins and outs of using Sample Wrench’s Enable scripting language. Enable is Visual Basic compatible and can be used for a variety of purposes including automating repetitive processes, batch processing of files, creation of alternate user interfaces, and even your own custom plug-in functions.

Starting Sample Wrench

From the Start menu:

Select Wrench from the Start menu bar. Once Wrench has started, have fun.

From the Desktop:

If you have used the default install, there will be a “dissidents” directory inside of your “Program Files” directory. Inside “dissidents” will be a “Wrench” directory, and inside of that will be the “Program” directory where you will find Wrench itself.

Double click on the Wrench program icon to start the program. Once Wrench has started, have fun.

Tutorial & Overview

Simply put, Wrench is designed to let you manipulate and alter sounds to your desire. In short, Wrench lets you see what your ears hear. It is to the world of sound, as a desktop publisher is to words and images. With the flexibility and power of its many functions, the resulting sound may be so altered that it becomes a new sound in its own right. Wrench comes in two flavors: one that stores sound data in 16 bit integer form, and one that uses 32 bit floating point form (Wrench 24/96). To you the user, both versions appear and operate in pretty much the same fashion. The differences are internal. The first version uses a 16 bit representation of the sound, just like music CDs. It offers very high quality editing with a signal to noise ratio and dynamic range of up to 96 dB. Wrench 24/96 uses 32 bit floating point values. Although the memory requirements are doubled, the signal to noise ratio can be greater than 140 dB, and the dynamic range can be over 190 dB. The sound data maintains a minimum 24 bit resolution at all stages. Mind you, there are no sound cards which can match these specs, although converters in the 20 bit range are becoming more popular. 32 bit floating point data ensures that your sounds are always treated with kid gloves. In fact, many of Sample Wrench's internal calculations use 64 bits for best results. One obvious difference between the two versions is that Wrench 24/96 allows signals to be over-scaled. In other words, signals can go over clipping level and not be damaged. Over-scaled waves will be drawn as though they are clipped, and they'll even sound clipped if you preview them. These serve as important reminders since you don't want to save your sounds over-scaled. The huge advantage is that you can rescale the waves by reducing their gain (volume level). Since the wave really wasn't clipped in the traditional sense, you can rescale to get back a nice undistorted wave. (The **Maximize** function is particularly good for this purpose). This can save a lot of time since you won't have to gain scale waves prior to certain functions (like EQ) in order to avoid clipping. Since both flavors of Sample Wrench operate in essentially the same way, we'll refer to them both as simply "Wrench". Where there are specific operational differences, we'll point them out.

Presently, there are a number of sound cards and MIDI samplers available to the musician. Generally, they all do a good job of capturing sound in a digital form, and playing it back. Unfortunately, most of them lack either a simple and easy-to-use user interface or the capability of significantly altering waveform data. This is kind of like having a V-12 racing engine run on only 5 cylinders! Wrench can be used with any standard sound card. Many of these cards contain very simple editors without a decent number of effects or features. Wrench offers a prodigious array of useful and unique sound processing tools. It will truly extend the usefulness of your sound card and offers you considerable flexibility in creating or editing sounds. When accessing an outboard sampler, the basic use of Wrench is in sending the wave data to the computer where it will be modified, and then sent back to the sampler. In this way, you get to do things to the sound that the sampler can't do, and, you get to do it in a very friendly environment. Even if your sampler does sport some of Wrench's features, you will probably find Wrench to be more informative, and thus, more efficient. Perhaps just as important is the ability to crossload sounds. Since Wrench talks to many different samplers and stores waves in a common form, it is very easy to send sounds from one sampler to another. Also, sounds may be transferred to/from sound cards and MIDI samplers via Wrench. Finally, sounds may be transferred to/from other platforms since Wrench supports a variety of file formats.

Wrench is a RAM-based sample editor. This means that it loads sound files into the computer's main memory (RAM) from disk rather than operating directly on the disk files. This approach has certain advantages. RAM is much faster than disk drives are, and thus, you get the fastest possible performance. Also, you are never working directly on original source material, so if you foul up an edit, you haven't lost the source. Interestingly, sample size is not limited by available RAM. Thanks to virtual memory, portions of your hard drive can be used for larger-than-RAM sound samples. In this way, you get the best of both worlds. The only cost to you is that once an edit session is finished, the sound sample must be saved back to disk. (If you forget, Wrench will warn you about closing an editor that contains a wave with unsaved edits).

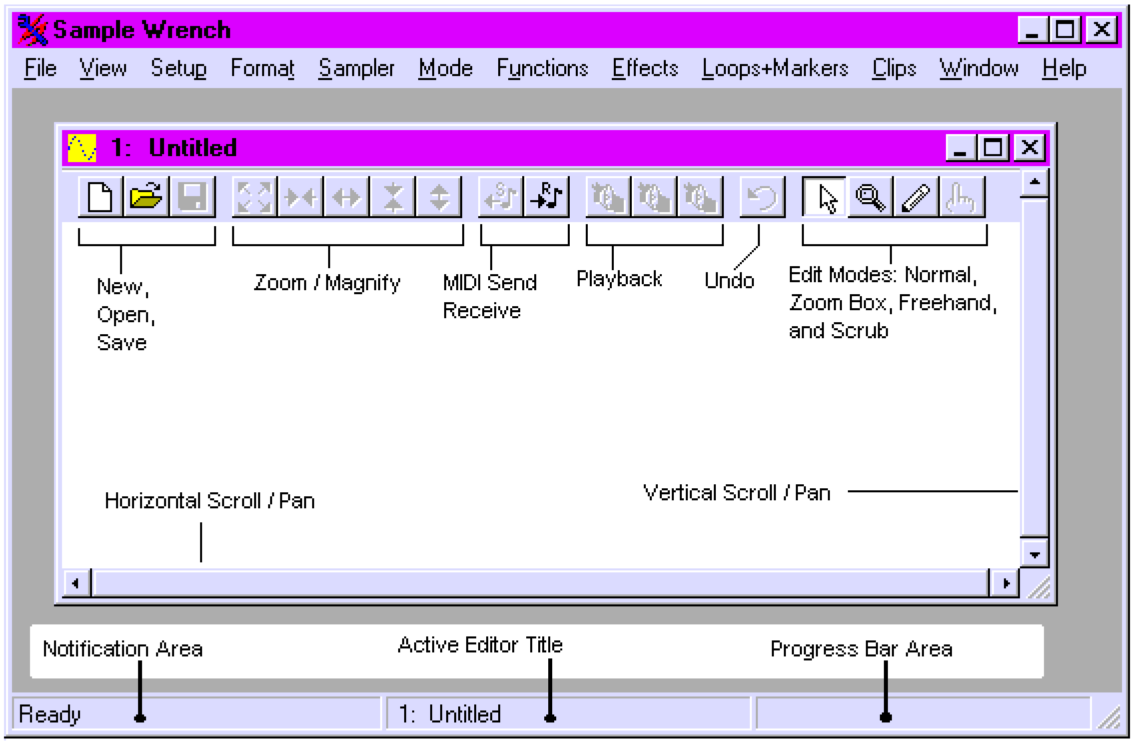
Let's take a closer look at how some of this is accomplished with an example. First, start up Wrench. Once Wrench is up and running, you should see a window with a title bar at the top. This is your background work area. It is here where you'll start the editors and eventually quit the program. You will also see a smaller window contained within it. This second window is an editor. Individual waves are loaded into editor windows and you can have up to 99 editors open at once. (More on the editors in a moment.) Along the top you will see a list of menus. They include **File**, **Setup**, **Format**, **Sampler**, **Window** and **Help**. These are the general menus. There are also a bunch of items specifically for editing chores. These include **Edit, View**, **Mode**, **Functions**, **Effects** (FX), and **Loops + Markers**. Let's look at the general-purpose menus first.

The **File** menu allows you to open each of the editors, open and save sounds, control playback and recording, initiate sample transfers, and quit Wrench. The **Setup** menu has many uses, one of the more important ones being deciding which segment of the wave is affected by edits. Besides selecting which channel of a stereo wave is to be affected by edits (a mono wave is considered to be sole "left channel"), you can specify which portion of a given channel is to be affected. You can affect the entire wave, the portion presently in view in the editor window, a range specified by a pair of markers, or a range defined with the mouse. For now, select **Setup/Affect All** and **Setup/Edit Left** (i.e., make sure that these items are check marked).

Under **Setup** you'll also find the choices Backups, Edit and Play, Abortable, Auto Zoomout, Smoothing, Save/Load Config, Save/Load/Assign/Record Macros, Sounds/Presets Path and Toolbars. The Backups item works in conjunction with the editors' Undo/Redo feature. You are going to need the Undo feature, so **Backups** must be enabled (make sure that Backups is set to one level deep). The Config items allow you to save and load Wrench configuration files. These files are used to remember your operating environment (such as your paths, editor attributes, and so forth). The Macros items are for assigning, loading, creating, and saving macro scripts. The Path items let you set the default directories for your sound and presets files. Generally, the Setup items are set-and-forget. We don't recommend constantly changing these choices.

The **Format** menu determines how newly saved waves will be written. Wrench can load and save waveforms in a variety of formats. Note that Wrench is smart enough to figure out the type of file it is reading in, so the desired type need only be checked for file saves (the sole exception to this is when you're using the Enable scripting language to read in the RAW formats. These need the proper menu selection to be read in, since RAW files contain no internal descriptions). Also, note that it is possible to override the default format choice when using the Save As dialog.

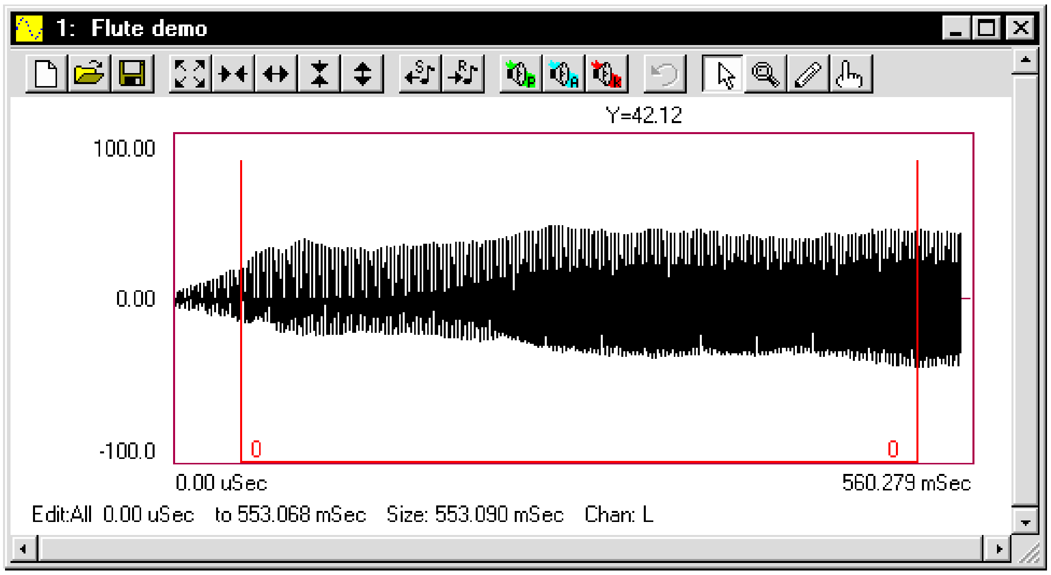
Moving on, the **Sampler** menu allows you to set a specific sampler. Next comes the standard **Window** menu that is used to arrange the various editor windows. The final menu is **Help**, which leads to on-line assistance. **Help/About...** shows the program version number and date, as well as our address. In order to keep things consistent for all users, the example will bypass the use a sampler and use an existing disk-based sound.



Since you would like to view and manipulate a sound, you must call up an editor. By default, Wrench opens an editor for you when first started. (If you need more editors later, simply select **File/New Editor** to start up another editor. The new editors look just like the first one except that they are numbered differently.) You will note that the borders contain small pushbuttons and sliders. These allow you to zoom in/out and pan left/right/up/down. The size of the window may be adjusted by grabbing and moving any of the edges. Note that the sliders are scaled along with the window. In the upper right corner you will find the standard window close, maximize and minimize buttons. When you are done with this editor, you may close it down by clicking on close (but not yet). Also, note that the center section of the main window's status bar (at the bottom) indicates that this is the active editor by showing the editor's number and sound name. You would like to load a wave from disk. Select **File/Open**. The standard system file dialog appears. Move to the Sample Wrench Sounds directory. The center area lists many of the available files. You may move through this list with the arrows and scroll bar. You may also preview sounds directly off of disk by selecting the file and then hitting Preview (8, 16, and 32 bit WAV files only). To stop preview in midstream, simply hit Preview a second time. Look for a file called **Flute_demo** and double click on it. The disk drive light should come on and a moment later, the edit window will contain a picture of the wave. The horizontal axis represents time, with the start of the wave at the extreme left. The vertical axis represents signal strength. This representation is called a time domain view, and is similar to what you would see on an oscilloscope.

The Ever-Changing Flute

Now that you have the flute loaded, you can start having some fun. First off, notice what happens when the window is resized; the wave is scaled to fit the new window. You can zoom in and out in both the horizontal and vertical directions by using the four double-arrow buttons in the top toolbar. These change magnification by factors of two. Notice that as you zoom in, the slider knobs get progressively smaller. The knobs indicate how much of the wave you are presently looking at. You may also pan left, right, up, or down by using the single arrow buttons next to the scrollbars/sliders (panning is often referred to as scrolling). These will move your viewing position about 10% of the present view size. Note that when you pan, the position of the slider knob will change, indicating which part of the wave is presently in view. You may also pan by dragging the slider's knob to a desired view location. By clicking in the slider area next to the knob, you can page through a wave. You will be looking at the next page or data frame of the wave. In other words, the right-most edge will now be on the left, or vice versa (likewise for top to bottom). Note that once you zoom in far enough, the wave will not be drawn in a filled in manner, but rather, like a single pencil line. When the wave looks like this, you're ready for microsurgery!



Edit:All 0.00 uSec to 553.068 mSec Size: 553.090 mSec Chan: L

Sometimes, you'd like to zoom in on a specific area, and the zoom buttons can be rather cumbersome. For this case, use the Zoom Box mode. Before using this, make sure that you are not already zoomed all the way in from using the Zoom In buttons. Simply click on the two Zoom Out buttons a couple of times. A very quick way of zooming out completely is to select the **Show Full** button (four diagonal outward arrows). To enter Zoom Box mode, hit the button with the magnifying glass on it (or select **Mode/Zoom Box**). The mouse pointer will turn into a small picture of a magnifying glass. The hot spot for this pointer is in the upper left (the reflection point). To use this, position the pointer near the area that you'd like to take a closer look at. Hold down the left mouse button and move the mouse over the area of interest (any direction). You should see an outline over the wave. This is the area that will be shown in the new view. Release the mouse button. In a moment the new view will be drawn. Before you proceed, return to Normal mode by selecting the pushbutton with the normal mouse arrow on it (or select

Mode/Normal). You now know the basics for moving around in a wave. This will be very helpful later on when you set loop points, do free hand drawing, or use any number of functions.

Previewing the Wave

Seeing may be believing, but the ears have final say in matters of sound. It is possible to use the computer's sound circuits to listen to the wave. Be aware that the quality of this sound may be greater or less than that of the associated samplers. To listen to the flute, first make sure that the audio outputs are connected and turned up to a moderate level. There are a few different ways of listening to sounds with Sample Wrench. The easiest way is to simply click on the loudspeaker button with the little **P** in it, in the toolbar of the editor window (P stands for Playback). This will play the sound with limited looping, assuming that a loop has been set. The playback can be aborted early bit clicking on the loudspeaker button with the **K** (Kill Playback) button. Feel free to play the flute! The button with the little **A** on it is for playing back the current edit Affect area (in looped mode, usually). This can also be achieved by hitting the Spacebar.

Another way to preview waves is with the Keyboard window. This is somewhat more advanced and is typically used to help with keymapping. If you're new to sound sample editing or don't use MIDI samplers, you might want to skip to the next page. To use the Keyboard window, select **Looping and Keymaps/MIDI Keyboard..** from the **Functions** menu. A window will open up with a 128 note keyboard. Three colored squares will indicate the root, high and low note settings for the wave. To play a sound, simply place the mouse over a key and depress the left mouse button without releasing it. The sound will begin to play. It will continue to play for as long as the button remains down, assuming that this is a looping sound, and not a one-shot type. To stop playback, release the mouse button. While the mouse button is depressed, you can hear new pitches by sliding the mouse along the keyboard. Playback will also stop if you move the mouse off of the keyboard. Before continuing, close the Keyboard window.

Now that you can preview sounds, you'll be able to hear the results of your edits immediately. Sound good so far? Things are only going to get better!

Maximize (Scale to Full) and Undo

Let's dive right into a few functions. One very useful function is called **Maximize (Scale To Full)**. This increases the wave to its maximum amplitude. Judicious use of this function can keep your noise floor at its minimum. Select **Functions/Level Control/Maximize**. Wrench will calculate just how much gain this wave needs for you, and then apply it. Since no further input is needed from you, no function dialog is displayed. While Wrench is figuring out the gain, the mouse pointer will turn into an hourglass symbol, indicating that the editor is busy. Wrench will also update a progress bar in the lower right corner of the main window. Once the calculations are completed, the new wave is drawn and the mouse pointer is reset. Also, the elapsed calculation time is given in the lower left corner of the main window. If you'd like, you may preview this sound by selecting the **P** button as before. Note that the flute sounds basically the same, it's simply louder.

If for some reason, you decided that you didn't like the results of this function, you could get back the previous wave by selecting the **Undo** button, which looks like a counterclockwise arrow (or by selecting **File/Undo**). Do this now. Note that the wave that existed right before the Maximize function was used has reappeared. It's as if the function was never used. For Undo to work, **Backups** must be enabled at startup (remember?) Note that the Undo can be undone. This will return the fully scaled wave. Do this by selecting Redo. Select Undo one more time to get back the original unscaled wave.

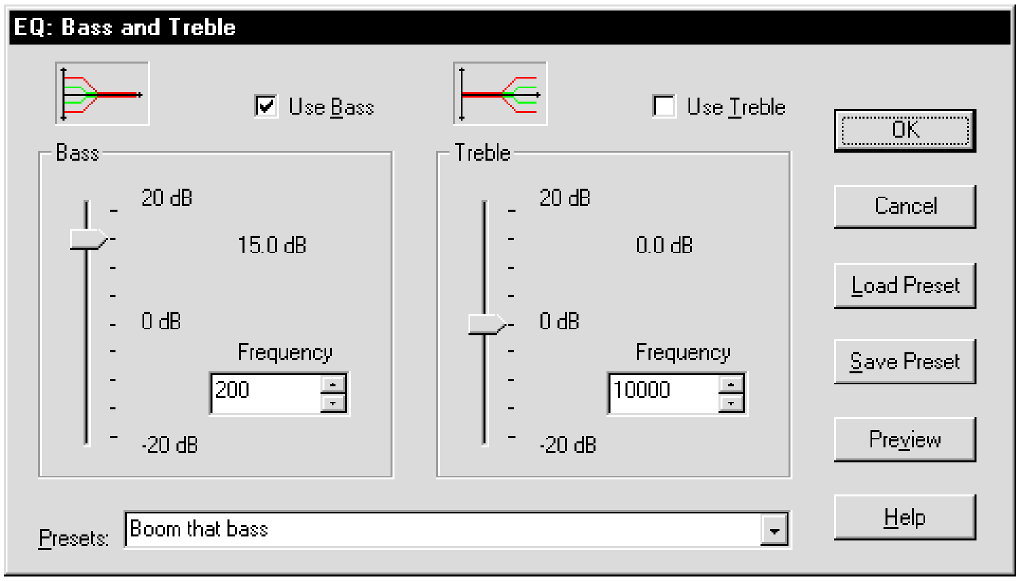
Digital Equalization

One of the more common signal processing functions used in day to day work is equalization, or EQ, as it is often called. This ranges from the simple bass and treble controls found in home stereo gear, up through to sophisticated parametric equalizers in use by studios and sound reinforcement professionals. EQ allows you to change the spectral balance or timbre of a sound. Wrench has extensive digital equalization.

There are four basic equalizing functions that can be used in Wrench. These functions are high/low pass filter, bass/treble shelve, multi-band adjustable graphic, and parametric EQ. As an example, you are going to add a moderate amount of high frequency boost to the flute wave. The simplest way to do this is via the bass/treble shelving control. To see the Bass and Treble EQ section, select **Functions/Equalization/Bass+Treble**. Note that the bass controls are on the left and the treble controls are on the right. This equalizer is more flexible than the average bass or treble control because it allows you to specify the frequency at which EQing starts to take place. For this example, use 2 kHz. Simply type the number

```
2000
```

into the **Treble Frequency** slot. (If the slot is not selected, just click on it first, then type in 2000). Now, in order to get just a treble control, make sure that **Use Treble** is checked and **Use Bass** is **not** checked. Once this is done, adjust the **Treble Boost/Cut** slider for a moderate amount of gain, say 3 or 4 dB. You can listen to what the result will sound like by hitting the **Preview** button. Many of Wrench's functions and effects have this preview capability. It is convenient since you can hear edits before committing to them. The previews are immediate and interactive (if you change parameters, the playback changes along with them).


EQ: Bass and Treble

[✓] Use Bass

[ ] Use Treble

Bass: 15.0 dB, Frequency 200

Treble: 0.0 dB, Frequency 10000

Presets: Boom that bass

To EQ the flute, click on the OK pushbutton at the side of the dialog. The mouse pointer will show the hourglass and the progress bar will advance as Wrench calculates the new wave. After a moment, the new equalized wave is drawn. Select the Play button in order to hear the new flute wave. The equalized flute should sound a little brighter than the original wave, and perhaps a bit breathy as well. (Note that excessive boost might cause the wave to become clipped and distorted, just as it would with a normal equalizer). The EQ section is very powerful and can alter waves in many ways. The Treble control is just one function of many. Before you proceed, select **Undo** to get back the original wave.

You will find many other processes besides gain, EQ and the like in the **Functions** and **Effects** (FX) menus. These include items such as amplitude compression, arbitrary envelope generation, time stretching, pitch shifting, echo, reverb, flanging, chorusing, and a host of others. We suggest that you play with these after reading the **Functions** and **Effects** sections. For now, let’s continue to another area of interest.

Markers and Looping

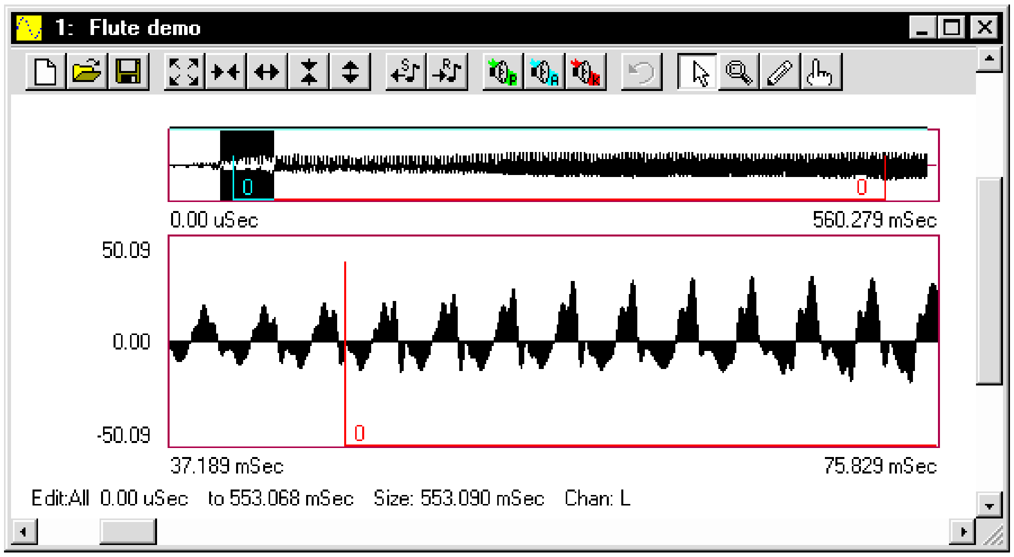
This section will be of particular interest if you need to create instrument or beat loops. Also, if you’re interested in creating a cut list or keep list, you’ll want to pay attention to the markers section since markers are used for defining the list segments. Markers can also be used to precisely define edit Affect areas. Even if you do not wish to follow this section step-by-step, we recommend reading through it anyway since it touches on a few important points.

If you have worked with samplers for a while, you know the importance of making good seamless loops. You probably also know the frustration of trying to achieve those ideal loops! Loops are a necessary evil if you want a sound to sustain over long periods of time (several seconds). Without loops, sustaining sounds would require tremendous amounts of memory. A loop is defined by start and end points within the wave. The audio circuitry in the sampler (or computer) plays the wave until it reaches the loop end point. When it gets to this position, the circuit immediately jumps back to the loop start point and continues playback. Every time the loop end is reached, the circuit jumps back to the start point. This goes on for as long as a key remains depressed. Visual editing is an absolute boon to the hardcore looper. By seeing what you're trying to loop, the whole process becomes much easier. Generally, you'll arrive at satisfactory loops in considerably less time. The trick is to visually determine front and back parts of the wave which are similar. By doing this, discontinuities in the loop are minimized, and thus telltale clicks and thumps are reduced. Some waves are easier to loop than others. Very clangorous or harmonically complex waves can be virtually impossible to loop perfectly. The flute wave is not particularly difficult.

In this section, you're going to create a new sustain loop for the flute wave since the default loop was not very good. At this point, you're interested in the **Loops+Markers** menu. First of all, loops are shown on the wave drawing as vertical lines connected by a horizontal bar. Loop ID numbers are drawn at the intersections of these lines. To see where your loops are, select **Loops+Markers/Loop View/Sustain**. You should see one line at the front of the wave and one at the rear. Only one loop is presently in use. It has been pre-defined to indicate the sustain loop. You can verify all of this by selecting **File/Properties (Info)**. Info will tell you the exact position of the loops (in samples). Wrench can deal with three different kinds of loops, a Sustain loop, a Release loop, and trial loops (for now, you only care about the Sustain loop). As you can see, this dialog provides other items of interest including the sampling frequency, wave size, and Marker locations. Markers are like bookmarks, and we'll take a look at them a bit later. Exit the Info dialog by selecting the OK button.

The reason why the default loop is so lousy is because the start and end points were poorly placed (on purpose). Zoom in on the start by using either the Zoom Buttons or Zoom Box. Magnify your view until only a few cycles (repetitions) of the wave can be seen. Try to memorize the shape of the wave and pay particular attention to where the loop line intersects (is it above, on, or below the center line?) Now, by using the horizontal scroll slider, move over to the loop end. (If you can't see the horizontal loop line on the display, you've gone too far, so backup). You will notice that the cycles in this section of the wave look a bit different than in the first part. Also, this loop point is intersecting in a different area. So, that's two good reasons why the loop sounds so bad! To fix this, you'll need to pick out similar areas. First, zoom out fully so that you can see the entire wave again. (Select **View/Show Full**). The loop start was placed on the attack portion of the wave. Usually, this is not a good place to start a loop because the sound has not yet reached full volume. Fortunately, the loop end is reasonably placed.

What you would like to do now is reposition the loop start. You can do this by selecting **Loops+Markers/Loop Set** and typing in a position from the keyboard. **A more direct way** is to use the mouse to grab the loop start line and move it to where you want it to be. Do this by positioning the mouse over the vertical loop start line, where it intersects with the horizontal loop line. Press the left mouse button. When you do this, the mouse pointer turns into an Insert pointer, and a highlight line is drawn. As you move the mouse, the highlight line moves with it. This line represents where you'd like the loop start to be moved to. We would like to skip over the initial attack portion, so move somewhat to the right of the original position and release the mouse button. The wave will be redrawn, showing your new loop start.



Edit:All 0.00 uSec to 553.068 mSec Size: 553.090 mSec Chan: L

This positioning was rather broad. Although you are in the right area, you don't know if the precise intersection is that good. Usually, loop points are set at zero-crossing points on the wave. You would like to do this. Zoom in on the loop start until only a few cycles are visible. Chances are, it will not be on a zero crossing. You will have to fine-tune your position. For the sake of convenience, look for a positive going zero crossing (i.e., a point where the wave intersects the centerline as it rises). Grab the start line with the mouse as outlined previously, and move it to the desired location. Before proceeding, note the general shape of the wave in the vicinity of the loop point. By using the horizontal scroll slider, move to the loop end. You will probably have to reposition this one a bit as well. Try to find a zero-crossing area that looks like the one you just did. Reposition this point as you did the previous one. Once this is accomplished, the loop is done. You may audition your work by selecting the P button as you did before. If you made reasonable choices, the loop should be much smoother than the original.

Successful looping is half art and half science. Do not be dismayed if this first attempt was less than perfect. As time progresses and you become more comfortable with Wrench and waves, your looping skill will increase. To make your life easier, Wrench offers certain automated looping features as well various forms of crossfade looping for those more difficult sounds. Of primary interest is the **Interactive Loop Window** item found under the Functions/Looping and Keymaps menu. Selecting this item will open a new window that shows the start and end portions of the loop. You may zoom in on this display using the window buttons as you would in an editor window. The difference here is that you have two sets of left/right arrows. These arrows control the positioning of the loop start and end points. You may select one of two views: **Simultaneous**, or **Spliced**. Simultaneous places the loop points in the middle of each display (loop end on the left, loop start on the right). Spliced places the two loop points between the two displays. In this way, you can see what the effective waveform is, as if you had spliced the portions together. (Ah yes, memories of magnetic tape and razor blades). A good loop will have a seamless (i.e., non-abrupt) splice. We'll take a closer look at this function a bit later. Make sure that the Loop Window is closed before you continue. As a side note, even though you only fiddled with one loop, you may define up to 256 different loops using Loops+Markers/Loop Set.

For details on creating, managing, editing and other advanced uses of markers and loops, refer to the **Markers, Loops, and Clips** section later on in the manual.

Free Hand Drawing

Well the looping tour was rather involved, so let's turn to something a little more, shall we say, artistic (but feel free to make a pit stop at the fridge first..). That little something is called Free Hand Drawing. It's kind of like painting with sound. You can draw arbitrary waveshapes! As an extreme example, you could draw an entire sound by hand. This is no small feat, though, and is best left to lunatics, millionaires, and other people with lots of time on their hands. Usually, Free Hand Drawing is used to smooth out discontinuities due to cut-and-paste splicing operations. It can also be used to smooth out clicks or other extraneous impulse noises from source material. (Please be advised that sampling sounds from albums and CDs is normally a copyright infringement, and as such, is against the law. Please respect the artist's rights). In this example, you can just have a little fun with it, and forget about being serious for a moment.

To use **Free Hand Draw**, select the button that looks like a pencil (or select the **Mode/Free Hand Draw** menu item). The mouse pointer will turn into a pencil. In order to draw with the pencil, the wave must be magnified to the point where it is no longer filled in. Trying to draw at a lower magnification could destroy the wave, so Wrench automatically disables the pencil for you. Once you've zoomed in far enough, place the pencil over the wave. Press the left mouse button and sweep the mouse from left to right as if you were writing. Notice how the old wave is overwritten with one from the pencil. When you release the mouse button, drawing stops and the exact shape of your new wave segment is calculated and redrawn. As long as you are in this mode, you can keep pushing the mouse button and drawing! If you move from right to left the pencil will erase the preceding wave segment, so that you can redraw over the segment without releasing the mouse button. If you decide that you don't like what you just drew, move back over it to erase it and then release the mouse button. By doing this, Wrench will ignore the drawing and replace it with what you started with. As with all of Wrench's functions, the Undo feature works on Free Hand Drawing as well. You may find the backward erase action of the pencil to be quicker, though. Before you leave this section, return to Normal Mode by selecting the Normal Mode pushbutton that looks like a regular mouse pointer (or by selecting **Mode/Norma**l). The pencil pointer should be replaced by the standard pointer.

Cut, Paste, and Clip

Well, it's time for a popular element of the tutorial. Even if you're an old hand using cut, copy, and paste, you'll probably want to stick around and read the section on using Wrench's Multi-Clipboard. It extends the concept of the normal clipboard by allowing several clips to be present at once. It's quite powerful.

Before you start, make sure that you can see the entire wave (select **View/Show Full** or the Show Full pushbutton). In this final section, you shall perform cut and paste operations. This is kind of like tape editing with a microscopic razor blade! Functionally, it works a lot like the cut and paste operations found in many word processors. You can do some pretty neat things with these functions. You can lengthen or shorten sounds, splice different sounds together, or even rearrange different parts of a sound. For example, you might sample someone speaking a sentence, and then alter the order of the words. (As a gag, you might rearrange your favorite politician's latest speech and turn it into a pile of meandering, useless, gibberish. Don't be surprised though, if you can’t tell the difference).

Sample Wrench’s basic Cut, Copy, and Paste operations are found under the **Edit** menu. These utilize the system clipboard so sound segments can be imported or exported from/to other audio programs. Wrench also has its own extensive internal clipboard called the Multi-Clipboard. Multi-Clips are a bit more advanced so we’ll hold off looking at them for a bit.

In brief, **Copy** copies the edit Affect area to the system clipboard. **Cut** is similar. It removes the edit Affect area from the wave and copies it to the system clipboard. **Paste** replaces the edit Affect area with whatever is in the clipboard. Note that the system clipboard can only hold one sound clip at a time, so if you perform several Cuts or Copies in a row, only the last one remains in the clipboard (you guessed it; if you need multiple clips, that’s what the Multi-Clipboard is for, see below).

Although you can use any Affect mode you’d like, Cut and Paste operations generally work best if you define edit areas with the mouse. Select **Setup/Affect Mouse**. You can now define an edit area with the mouse. For this example, we’ll grab the beginning part of the waveform. First, position the mouse about one third of the way into the main waveform drawing (the wave starts on the left and progresses to the right). Now depress the left mouse button and sweep the mouse toward the start (to the left). You will notice that the area is drawn in reverse highlight. When you get near (but not past) the left edge of the drawing where the amplitude is small, release the mouse button. The area should remain in reverse highlight. This is your new edit Affect area. We’ll cut this from the wave, so select **Edit/Cut**. In a moment the wave will be redrawn. Note the rather abrupt change where the section was cut out. The highlighted section was removed and copied to the system clipboard. Hit the Play button to hear the newly edited wave. It probably has a bad click in it now! Normally, you wouldn’t just randomly remove bits of waves like this - we’re doing this just for demonstrative purposes. Since the original chunk presently resides in the clipboard, we can paste it wherever we’d like. We can even paste it in several different spots if we want. To paste the clipboard contents into our wave, we first need to indicate where we’d like the chunk to go. For this example, we’ll paste it into the back portion of the wave. Position the mouse near the back (right side) of the wave. Since we want a simple insert without replacing anything, just click the left mouse button once. A line will appear where you clicked. This is the insertion point. Now select **Edit/Paste**. The contents of the clipboard are inserted and the wave expands. To see the entire wave, select **View/Show Full**. If you wanted to replace an area with the clipboard contents, you’d highlight the area to be replaced by holding down the mouse button instead of just giving the single click for the insertion point.

That’s all there is to it! Remember, since all audio programs are using the same system clipboard, you can Copy something in Wrench and then Paste it into another audio program, and vice versa. Sample Wrench also allows you to “grab” parts of a sound with the mouse and move them around, rather like shuffling cards in a deck. For more details, see the section entitled **Shuffle** under Edit Modes.

A Quick Tour of Multi-Clips

If you’re an advanced user, you may be interested in Multi-Clips. If not, just skip to the next page. You can always come back and explore them later. The basic functions for this section are found under the **Edit/Multi-Clips** menu. Basically, the routine runs something like this: you define a Multi-Clip (a chunk of the wave) by using either Markers or the mouse, then you either

Cut it (remove it from the wave), or Copy it for later use. The copied clip is placed in a Clipboard that can be accessed from any of the editors. From the Multi-Clipboard, a clip may be Pasted into a wave (inserted into the wave at a desired point). The Multi-Clipboard can hold many clips, limited only by your computer's memory and disk space. This is a very important difference relative to the system clipboard that can only hold one audio clip at a time. To help keep things straight, each clip can be given its own unique name. This is the basic theme. There are several useful variations and features besides. Also, the Multi-Clipboard is common to all of Wrench's editors, so it's a convenient way of moving sound fragments between them.

First, you're going to grab a clip from the flute wave. Select **Multi-Clips/Clip**. A small dialog will pop up indicating that Wrench is ready to work on Clip #1. You can name each clip, so give this one a meaningful name like "fido". After you've typed

```
fido
```

into the **Name** slot, select the dialog's Mouse pushbutton. Defining clips with the mouse is very quick and easy (more exacting applications can use a pair of Markers or the area presently defined for edits). After the dialog disappears, the mouse pointer will turn into the placement pointer. You are going to grab the last half of the flute wave. To do this, position the pointer over the middle of the wave. Now, press and hold the left mouse button. While still holding the mouse button, move the pointer over to the right edge of the wave. Notice how the intervening area is shown in reverse highlight. This is the clip. Release the mouse button. You now have a clip called "fido" in the Multi-Clipboard. At this moment, the clip has not actually been copied to the Multi-Clipboard it is only referenced. To complete the job, select **Multi-Clips/Copy**. You might wonder why this is a two step process and why Wrench doesn't automatically copy the clip for you the way the system Copy function or many word processors do. Unlike their word processor counterparts, wave clips can be very large (several hundred thousand bytes or megabytes is not unreasonable). Unnecessary copying can slow things down and eat up your free memory pool. If you just wanted to Cut a chunk of the wave, Copy would not be required. Anyway, it would be nice to verify the clip. You can do this by selecting **Multi-Clips/Edit**. This brings up the Multi-Clipboard dialog. It shows each of the available clips, their names, and their sizes. The letter **C** after the clip number indicates that the clip has been Copied. Your Clipboard should contain a single clip called "fido". It should have a C after the 1. You just wanted to verify this, so return to normal processing by selecting the OK pushbutton.

At this point, you could either Cut this clip, or Paste it. Select **Multi-Clips/Cut**. This will remove the clip and your new truncated wave will be drawn. Time to try the Paste function. Select **Multi-Clips/Paste**. Wrench will need a paste position for this clip. The position may be specified by either a marker or the mouse. As usual, select the dialog's Mouse pushbutton. You will be greeted with the familiar placement pointer. It works just like the other placement pointers. Position it where you would like the clip to be inserted (just about anywhere is fine for this example) and press and release the left mouse button. After a moment of calculation, the wave is redrawn with the clip inserted. Once the initial clip has been copied into the Clipboard (as you've done), it can be pasted repeatedly (try this). Once you are finished with a clip, you should discard it by selecting **Multi-Clips/Erase** (do this and answer Yes at verification).

The one thing that you should bear in mind is the concept of the *Active Clip*. Remember, unlike the normal system clipboard, the Multi-Clipboard can contain numerous clips and thus you need some convenient way of indicating which one you want to use. The Active Clip is the one that is currently selected from the list. It is the one that will be Cut, Pasted, Copied, Replaced, or Erased. By default, the Active Clip is the one most recently created or used. You can change the Active

Clip by selecting Multi-Clips/Edit. Once the Multi-Clipboard dialog comes up, simply click on the name of the clip that you would like to become the Active Clip. If you don't remember what a given clip sounds like, select it and hit the Play button. You can also load sound files directly into the Multi-Clipboard, or save clips directly as sound files by using Load and Save, respectively. Further, you may also delete clips by using the Delete button. This is faster than setting an Active Clip and then selecting Multi-Clips/Erase. If you want to dump the entire Multi-Clipboard, you may do so by selecting **Multi-Clips/Clear All**. Since Erase and Erase All are destructive operations, you will be asked for verification. For details on swapping clips from editor to editor and other advanced uses, refer to the **Markers, Loops, and Clips** section later on in the manual. The capabilities of wave processing with the Multi-Clipboard are limited only by your imagination. At the very least, it can be used to just trim waves. On the other hand, bizarre new waves can be created through the combination of components from a variety of sources. Have fun with it.

Cleaning Samples

Now we're going to look at something a little different. In this section we'll concentrate on cleaning up a sample which is badly mangled with hum, pops, clicks, and quantization noise. We'll be able to resurrect a pretty nice sample by the time we're through. Mind you, there's no excuse for not recording sounds correctly the first time, but sometimes we have to make the best of what we're given. Before you continue, make sure that **Setup/Affect All** is selected.

We'd like to keep the flute sample around for later use, so we're going to open a second editor. Select **File/New Editor**. When the editor pops up, select **File/Open**. From the Sample Wrench Sounds directory, select the file **clean123.wav**. Once loaded, you'll see three major chunks in the editor. This file is just someone saying the words "one two three". Preview the wave. You'll notice that it's very low in quality. It contains a background buzz or hum, a few clicks and pops, and some nasty 8-bit quantization noise that is particularly apparent during the words themselves. This is a recordist's nightmare. We're going to deal with all of these.

First off, let's attack the clicks and pops. Select **Functions/Remove Clicks and Pops**. This is a pretty straightforward dialog box. The pops are not extremely nasty here, so select **Normal** and hit OK. In a moment the waveform will be computed and redrawn. Preview the new wave. You'll note that the pops have disappeared. Wonderful! The hum and other noises still linger, so we're only halfway there.

To get the remaining garbage out, select **Functions/Reduce Noise**. There's quite a bit here and to get the best use of it you'll need to read the section on **noise reduction**. For now, we'll just use a few typical settings to get an idea of what's possible. Just a couple of points: Noiseprints are used to remove unique sorts of noises such as hums, buzzes, and the like. Thresholding is used to remove noise that is spread out across a wide range of frequencies, like hiss. First, select the **Noiseprint with Thresholding - Aggressive** preset from the drop-down list at the bottom. This is going to be a little too aggressive for this wave, so move the **Threshold slider** to **-55** (instead of -45). Also, select **Clean Me!** from the **Waves list** at the top. You'll note that we've selected the very wave we're working on as the source of the noiseprint. This works well because the beginning of this wave contains nothing but the hum we're trying to remove. Wrench will analyze this starting section and use it to suck the hum out of the rest of the wave. The Threshold setting will be used to remove the quantization noise from the words. To start the process, select OK. Depending on the speed of your computer, this may take anywhere from a dozen or so seconds to several minutes to complete. As you can imagine, Wrench is doing a lot of work here. It keeps you informed of its progress by advancing a small bar in the lower right corner of the editor. When the process completes, preview the wave. You'll note that the result is very clean! It's a far cry from what we started with. You might notice that the voice is lacking a little edge or bite. If this is objectionable, it can be at least partially compensated for with some EQ. You might try about 3 dB of boost at 3 kHz using a 1 octave wide parametric.

Restoring damaged material is extremely useful, but let's be honest, lots of folks just like to make weird noises! If *audio warpage* is your cup of tea, you'll love the next section of the tutorial.

Warpage 101

This section is for folks who just like things *weird*. For die-hard fanatics, this is just the tip of the iceberg in terms of how weird Wrench can make things. We're going to look at two effects here, namely Impulse Modeling and Spectral Warp. If you've been following along, you probably have two editors open; one with the flute_demo sound, and the other with the nicely de-noised clean123 sound. You're going to need both of these for the following section. Although it's not critical if you properly de-noised clean123.wav, it is important that you have a good flute_demo.wav. If you've chopped and pasted flute_demo into something wacky, it'll be easiest if you just reload flute_demo from scratch. (select File/Open. Don't save the current wave just delete it. When the File Open dialog pops up, double click on flute_demo just like you did at the start.)

We're going to start with **Impulse Modeling**. This is an extraordinarily neat function. It goes by several names including acoustics modeling and ambience modeling. Normal people use it for creating realistic reverb, but we're going to do something a little different. Here's the deal: Usually an impulse describes how a sound is smeared over time so it's perfect for reverb effects. There's nothing that says that an impulse must represent the acoustical reflections in a room. An impulse can be anything. The nature of the impulse will be imparted onto the signal creating a new signal. We're going to use the flute sound as our impulse. Before we can use the flute, we'll have to reduce its level since it contains far too much energy right now. Select the **flute editor**. Reduce the level by selecting **Functions/Level Control/Gain**. Set the gain to **-30 dB** and hit OK. You should now have a very quiet flute.

Select the **clean123 editor**. Select **Effects/Impulse Modeling**. In the **Waves list**, select the **flute** for the impulse. Under **Optional Processing**, select **Fade Out**. Hit OK. In a moment, you'll get a new wave. When you preview it, it sounds like someone with a flute caught in their throat. Some folks say that this effect reminds them of a vocoder, or "talking synthesizer". It's not a vocoder, but it's interesting none the less. Of course, there's nothing that says you can only do this to voice; you can just as easily use two instruments on each other.

At this point a word of enlightenment is in order. If the impulse is too large, the resulting sound will be way over-scaled. In a normal editor such as the 16 bit data version of Wrench, this means nasty clipping, so you must reduce the impulse's amplitude before use. This can take a little guesswork to get it right. On the other hand, Wrench 24/96 uses floating point data with a much wider dynamic range. Even if a signal is over-scaled, that doesn't mean that it's clipped. You can safely reduce the level **after** processing, either through reducing Gain or using the Maximize or Normalize functions!

OK, let's remove the flute from the guy's throat by hitting **Undo**. Now it's time to do some further mangling. Select **Effects/Spectral Warp**. This function produces interesting pitch sweeps and harmonic/formant alterations. Let's keep it simple and select the **Being sucked into a black hole** preset at the bottom. Hit OK. In a moment, the new wave will appear. You can decide for yourself whether or not that's what being sucked into a black hole *really* sounds like.

Final Helpful Items

You may have noticed one menu area which has been largely ignored up to now, and that's the middle to lower portion of the **View** menu. These items do not alter the wave, they simply make it a bit easier for you to read and interpret them. You have probably noticed that the horizontal and vertical axes are calibrated. The defaults are seconds for the time axis and percentage of full scale for the amplitude axis. These can be altered via the Horizontal and Vertical items to display a variety of formats. Also useful is an **XY Readout**. By selecting this, the XY position of the mouse pointer will be reported in the upper left-hand corner of the window. **Color Point** is useful when you are working at extreme magnifications. When this is selected, each individual sample is highlighted. **Box Outline** draws a box around the working wave area. Its purpose is purely aesthetic. Use it if you like it, ignore it if you don't. **Overviews** allows you to see a second drawing of the entire wave at the top of your edit window. The area you have zoomed in on is shown on the overview in reverse highlight. Further, the area that has been defined for editing is shown as a thin, highlighted bar along the top of the Overview. Selecting **Edit Status** creates a line of info along the bottom of each editor that details the area that has been chosen for edits. With **Set Colors**, you can customize the colors used to draw the waveform and such . **Set Font** brings up a standard Font Dialog. The selected font will used for the waveform labels and axis values. It is suggested that you stick with simple mono-spaced fonts of modest size in order to maximize the waveform drawing area. The **Set Offset** item allows you to have the time axis offset by a particular amount, which can be useful if you're "flying in" segments of audio. It also serves as a seconds/samples/SMPTE frames calculator. Of final interest are the **Get View** and **Set View** items. Each sound can have up to 10 auxiliary views. These can be useful if you're constantly bouncing between a few different areas of a large waveform. To use them, zoom and pan to a desired magnification level and portion of the wave, and then select one of Set View items. You can now change magnification and position, and immediately revert to the original view by selecting the corresponding Get View item at a later time.

Closing Down the Tutorial

To close an edit window, click on the **Close** button in the upper right corner. Since this operation will forget the contents of the editor, you are asked for verification. After answering Yes, only the background window remains. Select **File/Exit** to leave Wrench. For a quick exit, you can select File/Exit from any of the open editors without closing each editor first. This will automatically close the editors for you.

Well, that wraps up the tutorial. At this point, you've been introduced to Wrench and have touched on a few of its many functions. Most of the major concepts have been covered and you should be able to fill in many of the details on your own. Wrench is very consistent in how you work with it. Fear not though, the next section gives the nitty-gritty details on everything! You may wish to experiment further with some of the supplied waves before you continue with the next section. This will allow you to get a tad more comfortable with Wrench. Remember, make sure that you only use your backup disks. And if you haven't sent in your registration, now is a good time to do it!

Fundamentals

Main Menus

Global settings are accessed via the initial main menus. These settings apply to the entire program, not just a given editor window. The background menu choices are **File**, **Setup**, **Format**, **Sampler**, **Window**, and **Help**.

File Menu

The first item under the File menu is **New Editor**. Selecting this will open one of the 99 available editor windows (more on this in the next section entitled **The Editors**).

When Wrench is run and no editors are open, the only other item under File is **Exit**, which ends the program. If an editor is open, the File menu will be expanded to include more items (see the section entitled **The Editors**).

Setup Menu

The Setup menu contains many items which control the general behavior of Wrench. This is where you define the edit Affect type, load and save config and macro files, and open toolbar windows.

The **Affect section** has four menu choices: **All**, **In View**, **Markers 0,1**, and **Mouse**. These items select which portion of a wave will be edited. If you wish the entire wave to be affected, select **All**. Alternately, once you have zoomed into a wave and are only looking at a small portion of it, select **In View** to affect just that portion. For example, you may have zoomed in to where you are viewing just the first third of the wave. If In View is active and you then use the digital filter function, only the first third of the wave will be filtered. If All was active, then the entire wave would have been filtered even though you are viewing just the first third. The third variation is **Markers 0,1**. Markers are, in essence, like bookmarks. Wrench supports up to 256 markers, using marker IDs from 0 through 255. You can use markers number 0 and 1 to specify the range of the wave to be edited. This can be very handy if you need to create and easily control precise edit areas regardless of your view. The final choice is **Mouse**. This choice allows you to define the edit Affect area by sweeping the mouse over the area of interest. Simply position the mouse over the start of the section, press the mouse button and then sweep the mouse over the area. To finish, release the mouse button. When using the Mouse choice, the display will show the edit Affect area in reverse highlight.

Since Wrench supports the editing of stereo waves, you can choose which of the channels will be edited using the **Edit Left** and **Edit Right** items. If both items are chosen, then edits will be applied to both channels. If neither item is chosen, Wrench will not affect either channel (and it will tell you that it has no work to do, as a sort of kick in the pants). Mono waves are treated as a sole left channel, so if you have a mono wave and only select Edit Right, Wrench will again have no work to do.

Edit and Play is for editing convenience. If this item is checked, then the sound will automatically be played back once an edit process (such as EQ or Reverb) is completed. This also makes a handy signal if you are editing something in the background (ie, while running another application).

Most functions can be aborted in midstream. To activate this capability, select the **Abortable** menu item. When an edit function starts, a small "Abort" box opens on top of the editor window. Clicking on this Abort box will abort the process. In most cases, what you end up with is a partially edited waveform. You can then get back the original by using Undo, as long as you have Backups selected. In some cases, you will get back the original waveform after performing an abort, so an Undo is not required (this occurs with

Resynthesis and Sample Rate Transpose). The following functions cannot be aborted, either because they don't change wave data, or because they simply move waveform blocks around: FFT, Remove, Replicate, Trim and all Clip functions (Cut, Paste, Replace).

The **Auto Zoom Out** item is for viewing convenience. Sometimes, editing a wave can make it longer or shorter in time (for example, when using cut and paste operations). If Auto Zoom Out is active and an edit causes the wave to change in size, then the view is adjusted to show the new wave in its entirety.

No one's perfect and humans make mistakes, that's why Wrench has an **Undo** feature. In order to use it, some form of backup must exist. By default, Wrench uses one backup. Without backups, the program uses less memory and runs a little faster. Of course, if you make a mistake, you're out of luck! In order to be able to erase those inevitable boo-boos, you need to select **Backups**. You can specify exactly how many levels you'd like to use, meaning how many processes can be undone. This uses available system memory. It is very fast and efficient. It also has the added bonus of allowing you to undo an undo for comparison purposes (this is called redo). This is ideal if you have a large amount of RAM (if not, then the backups will go into virtual memory which is somewhat slower). Backups are updated each time the content of a wave changes.

Many functions have the capability to perform a smooth transition edit. What this means is that if you edit a certain part of the wave, the beginning and ending of the part can be blended into the non-edited part, creating a seamless transition. For example, if you choose Gain and give one section a 6 dB boost, instead of there being a sudden transition from normal to 6 dB boost, the transition will be drawn out, making it less noticeable. You set the transition time and whether you'd like smoothing on the start, end, or both sections of the edit by selecting the **Smoothing** menu item (for no smoothing, make sure that both Start and End are unchecked). Smoothing is available for most functions and effects. If the transition time specified is longer than the editing area, the time will automatically be reduced to the maximum of one half of the edit area for you. The Smooth settings (Start, End, and Transition Time) are saved with config files.

Wrench allows you to save your working environment through the use of configuration files. These files are accessed via the **Load Config** and **Save Config** items. Configuration files remember all of your global settings, such as file format, sampler driver, etc., and also the attributes of the last editor open. When Wrench first runs, it looks for a default configuration file called "wrench.config".

Wrench contains a Visual Basic compatible scripting language called Enable. You can create scripts to automate the operation of Wrench. The **Assign Macros** item brings up the Assign Macros dialog. Wrench uses function keys F2 through F12 to execute up to eleven Enable macros. The names of these script filenames can be entered in the provided slots. The **Load Macros** and **Save Macros** items do as their names suggest, and load or save sets of function key assignments. The **Auto-Record Macro** item creates a macro for you, by keeping track of the edits you make. This can also be used as a handy outline for your own custom macros. As with configuration files, Wrench also tries to auto-load a Wrench macro file. The default filename is "wrench.macro". Wrench also has a default macro which it tries to run at startup if it is available, called "wrench.macroinit". For more info on scripts and the Enable language, see the section on **Enable Scripting**.

Instead of having the file dialog open into the current directory when looking for files to load or save, and you can define a different path using **Sounds Path**. This is very handy if you keep all of your sound files in a directory other than where you keep Wrench itself. This default path is saved with config files. If a default path has not been defined, then the file dialog will open into the current directory. In a similar vein, **Presets Path** defines a n alternate directory for your functions and effects presets.

The final item, **Toolbars**, has several sub-items: Clips, Effects, Functions, Loops+Markers, and Views. These items allow you to open floating toolbar windows. These small windows contain a series of buttons which give you direct access to items found in the menus. You may find it easier to use a toolbar window rather than using the associated menu. All toolbar windows may be resized, and support tooltips. Also, note that toolbar windows are not constrained to the main Wrench window the way the editors are.

Format Menu

While loading a sound file, Wrench can recognize a number of different formats and automatically figure them out, including **WAV** in either 8, 16, or 32 bit versions, **AIFF** (a 16 or 24 bit IFF format which is also popular on the Macintosh and Amiga), **8SVX** (8 bit Amiga format), **AU** (Sun's format, and also Next's .snd format), **Sound Designer I** (a 16 bit format on the Macintosh), **Studio 16 Version 3** (a 16 bit format on the Amiga), and **VOC** (Sound Blaster). Wrench can also read a series of **RAW** types: 8 bit signed or unsigned linear PCM, 16 bit signed linear PCM in either Motorola or Intel format, 24 bit 3-byte-packed format, 32 bit floating point, and 8 bit u-law and a-law companded formats. If Wrench can't determine the file type, it will attempt to load it as a RAW format file. A Raw Open dialog will pop up, simply select the desired form and click OK. When saving a file, Wrench needs to know which format to save under. You set the default save format by opening the Format menu and then selecting the appropriate type from the list presented. You can also specify the file format from the Save As dialog box. For further info on file types, see the section on **MIDI and Disk Files**.

Sampler Menu

The Sampler menu contains a list of sampler types. Sending and receiving MIDI data dumps are initiated directly from the editors. The Sampler menu allows you to choose which sampler communication driver will be used. There is one for each different type of supported sampler. Since MIDI communication is so important to this program, separate sections of this manual (**MIDI and Disk Files**, and **Samplers**) are devoted to the topic of choosing MIDI drivers, and sending and receiving data dumps.

Window Menu

This is the standard Window menu. Your choices are **Cascade**, **Tile**, **Arrange Icons**, and **Close All**. **Cascade** arranges the open editors in a continuous overlapping form while **Tile** arranges them in block fashion. **Arrange Icons** adjusts the icons of minimized editors. **Close All** shuts down all open editors.

Help Menu

The Help menu contains the choices **Contents**, **FAQs**, **How-To**, and **About**. **Contents** gives a listing of all available help topics. The **FAQs** and **How-To** items are short cuts to popular sections. **FAQs** is a list of frequently asked question with answers, and **How-To** is a list of short, concise instructions on how to do various common operations such as opening and saving files, defining edit affect areas, playing sounds, etc. The final Help menu item is **About**. This displays the program version number and date, and where to contact us for technical support and other information.

The Editors

This section gives a general overview of an editor window and its toolbar, along with the File and Edit menus.

Sample Wrench allows you to process several waves simultaneously, each in its own editor window. Each editor has access to the same functions and has the same capabilities. Before you can work on a sound, the sound must be loaded into an editor. An editor lets you view a time domain representation of a sound. In other words, the horizontal (or X) axis represents time, while the vertical (or Y) axis denotes signal strength. The sound is read in much the same way as a sentence is read, that is, from left to right. The start of the sound is at the extreme left and finishes at the right edge. Wrench displays sounds using a very accurate true peak detecting algorithm. This ensures that no matter what your magnification level, you will always see a proper rendering of the wave. Wrench never skips over data points as some visual editors do. Because 99 editors are available, you can have 99 different waves loaded into your computer at the same time, ready for manipulation. This allows you to do things that are cumbersome or awkward (or downright impossible) on a simpler, single editor system. Since each editor operates independently of the others, and pretty much exists in its own world, we refer to them as virtual editors. Each editor has a number associated with it, found in its title bar.

Toolbar and Controls

You start up an editor by selecting **File/New Editor**. After the menu item is selected, the appropriate editor window will open. You can move or resize this window to fit your needs. All manipulation of this sound may only take place when this window is active (ie, title bar is not ghosted). All signal processing functions are accessed through the associated menus, while viewing position and scale may be set from the window border controls. Unlike the average window, each editor window contains extra controls along the top and the edges. Each border has a single slide control and two small buttons which set the view point. There are controls in the top toolbar to set the horizontal and vertical magnification. The simplest controls are the **Zoom In** and **Zoom Out buttons**. These have double arrows on them. Arrows pointing inward (at each other) indicate Zoom In. Clicking on these will magnify your view by a factor of two. The **Zoom Out buttons** do the exact opposite. Note that their double arrows point outwards, indicating an outward zoom or pull-back. Next to the scroll bars are Pan buttons. Clicking on these will move the display 10% in the chosen direction. If you click on the Pan buttons without releasing the mouse button, the Auto Scroll feature will kick in. As long as you keep the mouse button depressed, the wave will be continually redrawn, as if you rapidly selected a pan button several times in a row. At high magnification, the wave will glide by quickly, almost as if it's animated! This can be very useful if you want to scan through a wave to find peaks or other areas of interest. Finally, the toolbar also contains buttons to load and save files, play sounds, and initiate MIDI transfers. The **playback buttons** look like little loudspeakers. The one labelled **P** will play back the entire wave, while the one labelled **A** will play back the present edit Affect area. The **K** version is used to stop (kill) playback early. The **MIDI transfer buttons** appear as small notes with arrows next to them. The one labelled **R** is for Receiving sounds from a sampler while the **S** version is for Sending sounds to a sampler.

The slide controls are multi-purpose items. They allow you to move quickly to a desired section of a wave, and they provide nice visual feedback. You will notice that as you zoom in or out on the wave, the size of the slider's knob will change. The size of the knob indicates how much of the wave is presently visible. If the knob is as big as the slider, then the entire wave is visible along that axis (note that it is possible to have different amounts of magnification horizontally and vertically). If the knob is half the size of the slider, then only half of the wave is visible. The position of the knob in the slider tells you what part of the wave you are presently looking at. For example, if the horizontal knob is one third the size of the slider and is positioned at the left edge, you are viewing the first third of the wave. If this same knob is located midway, you are viewing the middle third of the wave. A quick scan of the two sliders will tell you right where you are in the wave. Besides visual feedback, the knobs provide a quick way of changing your viewing position. You can simply drag the knob to a desired location, and the view will be updated accordingly. You can also page through a wave by clicking on either side of the knob, inside the slider. Doing this moves you to the next page (or frame) of data. Paging is like panning, except that the jump is bigger.

It is very important to note that the sliders and buttons only change your present view of the wave. They do not change any wave data, and thus, the sound is not altered. They simply allow you to make close inspections of the wave, and make certain functions easier. Also, none of these controls will produce an effect if a wave has not been loaded. In other words, it is impossible to zoom in on an empty editor! Finally, editors may be closed down by selecting the standard close button found in the upper right corner of the window. If the editor is not empty, you will be asked for close verification. This prevents loss of the wave if you happen to accidentally hit the close button.

File Menu

The File menu contains the standard items **New**, **Open**, **Save**, **Save As**, and **Close**. **New** deletes the wave in the editor while **Open** brings up the standard file dialog so that you can load a new sound file. It also has the ability to preview sounds directly off of disk (8, 16, and 32 bit WAV files only- hit the Preview button a second time to stop playback in midstream). **Save** transfers the wave to disk using the present name. **Save As** also saves the wave to disk but allows you to specify an alternate name. The **Close** item removes the wave from the editor and then closes the editor window.

The **Send** and **Receive** items initiate MIDI sampler transfers. They are echoed on the editor's toolbar by a pair of buttons showing a musical note with an arrow. The Send button contains an **S**, while the Receive button contains an **R**. Selecting these items will bring up the Send or Receive dialog for the sampler selected under the Sampler menu. For more details, see the section on MIDI Transfers.

Sounds may be recorded directly into Wrench using your sound card, or you can generate certain waveforms mathematically. These items are accessed by the **Generate**, **Record Prefs**, and **Record** items. Further details on their operation can be found under the Recording and Generating Waves section.

There are four items on the File menu concerning playback. **Play Prefs** allows you to set your preferred playback parameters such as the device to use, the number of times a loop should repeat, a transpose value, whether or not you'd like a position tracking bar, and so on. The **Play** item plays back the entire waveform while **Play Affect** only plays the presently defined edit Affect area (see the Setup/Affect... menu for details on defining edit Affect areas). These two functions may also be accessed from the editor's toolbar. Look for the buttons containing small loudspeakers. The one with the **P** is the Play Entire button and the one with the **A** is the Play Affect button. The fourth item is **Stop Play**. This is used to terminate playback early. It is echoed on the editor's toolbar by a loudspeaker button containing a **K** (for kill play). Further details on playback may be found in the Playback and Preview section of the manual.

You may find the remaining handful of items in the editor's File menu to be quite handy, even if they aren't absolutely essential. For the most part, they are self explanatory and very easy to use. **Name** will let you change the name of the sound being edited. This is very useful if you have created a new wave and wish to give it a descriptive tag. The Name dialog is very small and contains only the text box and OK and Cancel buttons. To use it, just type the desired name into the text box and hit OK. The title bar of the editor will now show the new name. If you decide not to change the name, hit the Cancel button instead.

Another item of interest is **Properties (Info)**. Selecting this will present a listing of relevant statistics about the wave including its name, save path, size, sampling frequency, and file format (as it was loaded). The first six markers are also listed. In the middle of the dialog is a listing for the sustain and release loops. The loop start and end offsets are given, as well as the loop type (either forward or forward-backward). Keymap info is presented next in terms of MIDI note numbers. Fine tune and offset values are presented at the bottom of the dialog. Sizes and positions are given in sample numbers as well as in your presently chosen horizontal units (from View/Horizontal). To quit this dialog, just select the OK button at the bottom.

Towards the bottom of the File menu will be a list of recently used sound files. Initially, there won't be anything here, but as you use Wrench, the names of sound files will be added. Selecting one will load it into the editor if the editor is empty. If the editor isn't empty, a new editor will be opened and the wave will be loaded into it.

Edit Menu

The Edit menu gives access to a handful of common editing chores. The **Undo** and **Redo** items will let you move around recent edits, making comparisons easy. They will only be active if Backups is at least one level deep. By default, Backups are enabled. You can also get to Undo and Redo via the editor's toolbar- just look for the little counterclockwise and clockwise arrow buttons. respectively. **Undo History** shows a list of current editing steps. It allows you to move directly to a prior edit which may be several levels down. It effectively "unravels" all of the prior edits. It can also be used to move forward in a sequence. The current position in the list is highlighted. You can undo/redo to any other point by simply double-clicking on the edit you wish to go to. For further details, see the next section.

The **Cut**, **Copy**, and **Paste** items are used with the system clipboard. For details, see the section on System Clipboard. The **Delete** item is similar to Cut except that the removed chunk is not copied to the system clipboard. **Trim** removes everything outside of the edit Affect area while **Mute** silences the current edit Affect area. To avoid sudden turn-on or turn-off clicks, consider using the Smoothing feature (under the Setup menu) to automatically fade-out or fade-in to the silenced area. The final item is **Multi-Clips**. This leads to a sub-menu containing several choices. These allow you to manipulate the clips which you copy into Wrench's internal clipboard. Unlike the system clipboard, the Multi-Clipboard can hold many audio clips, each with their own name. For more info on Multi-Clips, see the section entitled Using Multi-Clips.

Edit Modes

There are four basic modes of operation in Sample Wrench. They are typically selected from the editor's **Modes** menu. They may also be selected from the editor window's toolbar. The first mode is called **Normal**. This is the default mode and where you tend to operate most of the time. It is a relatively safe mode in that the mouse buttons don't do much beyond the typical menu and selection functions. When you're in Normal mode, the mouse pointer is the default (arrow) pointer. In this mode, you have the ability to directly grab and move markers and loops using the mouse. To move a loop point, place the mouse pointer in the area where the vertical and horizontal loop lines meet. Hold down the left mouse button. If you grabbed correctly, you will note that a vertical highlight line is drawn at the mouse pointer. Move this to where you'd like the loop point to be, and then release the mouse button. Markers may be moved in a similar way. The "grab area" for a marker is immediately above or below its ID number (depending on whether its an even or odd ID, respectively). "Grabbing" also works in the waveform overview area, if overviews are activated.

The second mode is **Zoom Box**. It is used for quick area magnification. In this mode, the mouse pointer looks like a miniature magnifying glass. The hot spot for this pointer is the upper left corner. To use the Zoom Box, move the pointer near the area which you want to magnify. Now, press and hold the left mouse button. While still holding the button, move the pointer over the area of interest. You will note that a box is drawn around the wave fragment. When the mouse button is released, the area inside the box will be magnified and redrawn so that it fills the entire edit window. You can repeat this process as required. Also, it is possible to drag the outline box in any direction - you don't have to sweep from left to right or anything like that. The outline box will be automatically limited to the size of the active edit area if the mouse pointer moves outside of these bounds. Along with this, the initial box starting point must lie inside the active area. If you click outside of the active area, the zoom will be ignored. While in this mode, it is still possible to access all of the menus and use the window and dialog gadgets. Indeed, it is possible to run the entire program while in this mode (this is not recommended since accidental mouse clicks may produce zooms, and this can slow you down). As a side note, since Zoom Box only changes the present magnification of a wave and not the wave itself, the Undo function will not bring back the old view. In order to un-zoom (ie, zoom out), you may use either the Zoom Out buttons on the editor's toolbar, the Show Full function (found under the Sample menu), or an Alternate View. Zooming also works in the waveform overview area, if overviews are activated. This can be very handy for jumping around in a wave.

The third mode is **Free Hand Draw** (some people call it the pencil tool). Like the Zoom Box mode, all menus and functions remain available to you. Free Hand Draw lets you draw desired waveform segments by hand. It is particularly useful when splicing waves and clips together, or for removing impulse noises. When this mode is selected, the mouse pointer turns into a pencil. The hot spot for the pencil is the "lead" end. To activate the pencil, hold down the left mouse button. Moving the mouse from left to right while holding the mouse button will overwrite the original waveshape with your newly sketched version. When you release the mouse button, the exact wave data is calculated and then redrawn. If you move the mouse from right to left, the pencil will erase instead of write. When this is done, the sketch is ignored. This is useful if you make a mistake while drawing and would just like to return to what existed at the outset. It can also be used to repeatedly sketch over a particularly troublesome area.

Free Hand Draw mode will only be activated when the wave magnification is high enough. You know the magnification is high enough when the wave is drawn as a single line instead of being drawn filled in. This is done to protect you from inadvertently destroying waves when they are set for low horizontal magnifications (the sonic effect of this can be very nasty). For convenience sake, drawing is allowed at fairly low vertical magnifications. Be careful though, and always zoom in on areas that you've drawn. If you don't watch out, the wave may show a blocky, stair-step form. (If it does look like this, just draw over it again at higher and higher magnifications - each step makes it smoother). If the Color Point option (under Setup) has been activated, you will be able to see the exact results of your drawing. At very high magnifications, it may appear as though Wrench is ignoring some of what you draw. This is perfectly normal. Wrench can magnify the wave so much that only a few samples show on the window, and your drawing must be adjusted accordingly. In order to get a better feel for exactly how this works, go to extreme magnifications, enable the Color Point option, and try to edit just a couple of points. The Color Point will always indicate the real data. A helpful hint: drawing is limited to the active window area. If you need to start at the very beginning of a wave, the mouse placement needs to be exact. To avoid this, consider moving into the wave just a bit before pressing the left mouse button. You can then move the mouse left, to the beginning of the wave, and start drawing.

The fourth and final mode is **Scrub**. This allows you to hear the waveform using the mouse. To use it, select Mode/Scrub. The mouse pointer turns into a pointing finger. Now, move the finger to the area which you'd like to hear. Depress the left mouse button and move the mouse across the wave. When you release the mouse button, that section of the sound will be played back according to the direction in which the mouse was moving. Refer to the Wrench Play documentation for details.

Shuffle

While it's not a mode per se, if you're using mouse-based edit Affect areas (see the Affect section under the Setup menu), you can move selected blocks of audio directly with the mouse. We call this **Shuffle** because it reminds us of moving cards around in a deck. To use it, first highlight the desired area using the mouse. Grab the area by clicking on it without releasing the mouse button. Move the mouse to where you'd like the chunk to be moved and then release the mouse button. If you release it inside of the highlighted area, nothing happens (that would be like pulling a card out of the deck and inserting it back where it came from). For stereo waves, you can select individual left or right chunks by manipulating the outer portions of the waveform drawing (ie, the top half of the left channel or the bottom half of the right channel). If you manipulate the area where the two channels butt together (ie, the bottom half of the left channel or the top half of the right channel), you'll edit both channels simultaneously. Note that the mouse pointer will turn into a hand when you're Shuffling.

Wave Playback

There are basically two ways to listen to a wave: you can either listen to it using the computer's internal (local) voices, or you can send it back to a sampler and remotely trigger it from the computer. In the second case, the computer keyboard can be used as a musical keyboard. Final auditioning for samplers should be done directly from the target sampler with a high quality monitoring system.

There are several different ways in which you can preview a sound using the computer's internal voices; **Auto**, **Keyboard Window**, and **Scrub**. Some Functions and Effects dialogs also allow **Real-Time Interactive Preview.**

Playback Preferences Dialog

In order to set up the way the computer keyboard responds and other playback characteristics, Wrench uses the **Playback Preferences dialog**. In this dialog you will find check boxes and radio buttons for a variety of items. These are your system playback defaults. Selecting the OK button does not start playback, it simply establishes the settings as your present defaults. Normally, you set the Preferences at the start of a session and then leave them alone. Playback Preferences can be saved and loaded just like Wrench config files to save you more time.

Starting at the upper left is **Buffer Size**. The Buffer size section indicates the amount of RAM used by the computer to play sounds. Generally, smaller amounts produce quicker response, but require faster machines. Use the large size if playback sounds broken up. The **Channel** choice section comes next, and is mostly appropriate for stereo waves. You can choose either Left, Right or Auto. Auto will play whatever channels are available (mono sounds come out as dual-mono, the same sound from each loudspeaker).

Loop Choice affects sound files which have looping information (loop points are drawn as joined vertical bars on the display). Normally, sounds are played back as a one-shot. If the sound contains loop info, you can choose to override its internal specification and use either its release loop or treat it as a one-shot. The Auto choice uses whichever loop the sound file specifies (if any). Auto first checks for a sustain loop. If one isn't found, it looks for a release loop. If a release loop can't be found, then the sound is played back as a one-shot. The **Loop Direction** section allows you to specify the way loops are played back. Auto uses the specification from the file. You can, however, force loops to be played back in ordinary forward form, or in forward-backward form (ie, the sound plays forward from start to end, then plays backward from end to start, then forward again, and so on). Forward-backward loops are useful with certain types of musical instruments, but are not supported by all MIDI samplers.

Attenuation allows you to reduce the sound level by a certain number of decibels. This can be handy if you use Sample Wrench with other audio programs, so that you can match their output volumes. This setting does not affect the wave data itself, it only alters the playback volume for your sound card.

Selecting the **Show Position** checkbox creates a tracking bar which runs in sync with the playback so that you can see precisely where you are at any given instant. **Lock to 44.1 kHz** forces the audio output hardware to play back at 44.1 kHz, regardless of the sample rate or fine tune values of the wave itself. This is useful for the proper operation of certain audio cards (for example, to produce S/PDIF the card may require a 44.1 kHz playback setting). Wrench will recalculate the wave data during playback to achieve a 44.1 kHz rate without an associated pitch shift when you're using normal playback modes. Normally, this is not practical to do for effects previews since the resulting computations can be intense. Therefore, effects previews may experience pitch shifts if Lock is chosen and either the wave's sample rate isn't 44.1 kHz, or its fine tune isn't zero. **High Res 24 Bit** is useful if you have a sound card capable of greater than 16 bit resolution. Selecting this will use the high resolution playback capabilities for normal previewing. High resolution playback is often more taxing on the computer system and therefore realtime effects previews always use 16 CD quality playback. Due to its exacting nature, realtime preview using the ~~Loop Window~~ will use high resolution playback if desired. If your sound card does not have high resolution capabilities, you will be warned about this when you try to preview something. The **Transpose** slot allows you to shift the playback rate up or down in semi-tones (ie, half steps). The **Iterations** slot allows you to set the number of times a loop will playback. The **Driver list box** lets you select the audio driver you'd like to use (normally only used if you have several audio output devices available).

Finally, two buttons are available to save and load playback preferences settings. In this way, you can create different playback setups for different scenarios, and load them as needed.

Auto-Mode Playback

The Auto Mode has a "Hands Off" design. You enter Auto mode by clicking on the two **P** or **A** loudspeaker buttons in the editor window. The **P** button plays back the entire sound while the **A** button plays back the presently defined edit Affect area. For looping sounds, playback will continue until a certain number of loops have sounded (set by you). In either case, you can abort playback early by hitting the **K** (kill playback) button in the editor window. For the **A** button, the loop will be the entire edit Affect area, allowing you to hear the affect of edits easily. For the **P** button, the loop will be set by the **Playback Preferences** dialog. Also, note that Auto-Mode uses the **Fine Tune** value from the **Keymap dialog**, allowing you to hear small adjustments in pitch.

The **Playback Preferences Iterations** slot allows you to preset the maximum number of loops to be played back. If you want playback to continue for a long time, simply type in a large number, like one thousand or so. Remember, you can always break out early by hitting the K button.

You can hear the sound at a variety of different pitches, depending on what you set the **Transpose** amount to. The normal form is to use a Transpose of 0, which produces root pitch. in contrast, a value of +1 will shift pitch sharp one halfstep, while -2 will shift flat two halfsteps (ie, one whole step). It is possible to enter very large values for the Transpose amount. Wrench will attempt to play these back, but it is important to note the limitations of the computer's internal audio hardware. There may very well be various forms of distortion including aliasing. Also, extreme pitches (ie, large shifts) may not respond quickly to the K button.

By selecting the **Show Position** checkbox from the **Playback Preferences** dialog, you have the ability to "see" what you hear in realtime. In essence, a vertical bar will move through the display linked to audio playback. This makes it very easy to identify pops, noises, or certain segments of long sounds (such as one word out of a sentence). This is only applicable to the Auto type playback. This position bar will continue for as long as the sound plays. If you zoom in on the wave, the bar will continue as well. When you're zoomed in and the position is prior to what you're looking at, the bar will be at the extreme left. If the position is after the section you're looking at, the bar will be at the extreme right. You can zoom and scale while this is going on, if you wish. Be advised though, that horizontal scrolling may sometimes produce extraneous position bars. They don't hurt anything and you can get rid of them by simply resizing the edit window.

It is important to note that during playback, you do not have access to all menu items and functions. Basically, what you can't do is use the **Functions**, **Effects** or the **clipboard**. For example, you cannot play back a sound and simultaneously use the Equalizer. The reason is because playback can require a fair amount of overhead and this would slow down the functions. Since you can quickly launch playback in a variety of ways, you're much better off to perform the function and then audition it, instead of trying to do both simultaneously. Many things are allowed, though. When playback is active, you can call up most of the remaining menu items, the Loop Window, any of the items in the Loops+Markers or View menus, or change between Normal, Freehand and Zoom Box modes. If you change to Freehand mode, you can hear your changes as you draw them. This is particularly useful if you're trying to get rid of a click or smooth a splice.

By having access to the **Loops+Markers** menu, you can compare different sets of loops using the **Loops Set** dialog. For example, you may have two loops you'd like to compare. Let's say that their ID numbers are 2 and 5. First, set the SustainID to 2. Initiate Auto playback and listen to it. (Make sure that the Iterations amount is fairly large so the sound won't "run out"). While it is playing back, call up the Loops Set dialog again and change the SustainID to 5. When you hit the OK button, playback will immediately jump over to the values set by loop 5. If you delete the Sustain loop, playback will jump to the Release loop. If a Release loop has not been set, then the sound turns into a one-shot, and plays to its end. You can add, alter, and delete loops in this manner indefinitely.

During playback, you also have access to all of the normal waveform viewing, panning and zooming controls. On slower computers, waveform redrawing and mouse movement may be somewhat jerky. If the load on the microprocessor gets too heavy, Wrench will sense this and react by aborting sample playback.

Keyboard Window

The Keyboard window is used for three things: you can hear different pitches using the mouse, you can trigger a remote MIDI device, or you can set your keymap graphically. You open the keyboard by selecting **MIDI Keyboard..** in the **Functions** menu. There are two areas of interest in the window. The upper area is comprised of two claviers which together span the entire MIDI note number range of 0 through 127. The lower area is comprised of a set of buttons and other gadgets. The **Fine Tune** slider will show the present tuning, and the values for **Low**, **Root**, and **High note** will be printed next to their respective buttons. Also, these values will be reflected on the two claviers. Small colored squares will be placed on the keys to indicate the low, root and high values. The low note square will be placed at the bottom of the key, the root note square in the middle, and the high note square at the top. Normally, these squares will all be different colors making identification even easier (unless you're running with very few colors or in monochrome). You can change the keymap by using the mouse. To do so, select which of the three values you'd like to alter by clicking on the appropriate button. At this point the mouse pointer will turn into the word "To" as you pass it over the claviers. Now, click on the key you'd like the note set to. Once this is done, the text and colored squares will be updated to reflect your change. As a side note, if you'd like to set two or three of the values to the same key, simply click on the High/Low/Root buttons in sequence before clicking on the desired key.

In order to listen to a wave, simply click down on the key you'd like to hear. Playback will start and continue for as long as you hold down the mouse button (assuming that you have a looped sample and not a one-shot type). When you release the mouse button, playback will halt. You can listen to different pitches by simply clicking on different keys. If you move the mouse over the keyboard during playback, the pitches will change also. Note that the new pitches will not restart playback from the beginning. In this manner, you can listen to the effect of transposition on loops very quickly. This is also useful for listening to pitch shift effects on very long samples. Remember, you can always restart playback from the beginning by simply releasing the mouse button and selecting a new key. If you move off of the claviers at any time, playback will halt.

You can also direct the output of the claviers to a MIDI device instead of to the internal audio circuits. To do this, simply set the desired MIDI driver, volume and channel, and then select MIDI Triggers from the Clavier Output Type group.

Like Auto Mode, the Keyboard Window uses the Fine Tune value from the Keymap dialog, allowing you to hear small adjustments in pitch. You can adjust the Fine Tune amount directly and hear the result in realtime with the Fine Tune slider. To use this, simply click and hold on the slider's knob. This will start playback. As you move the slider, the Fine Tune readout will change and the pitch of the sound will start to shift. The pitch will increase as you move to the right, up to a maximum of 50 cents (one half of a halfstep) from the nominal. As you move to the left, the pitch will drop by a maximum of 50 cents. When you release the mouse button, playback will cease, and the last value on the slider will be used for the sound. If you decide that you don't want any shift at all, make sure that you return the slider to 0 cents. A word of caution: due to the inherent limits of the computer's internal audio circuitry, small shift changes may not be audible and there may be some lag in time response. The exact limits depend on the pitch of the sound and its sampling frequency.

Please note that the keyboard will allow you to play samples which are way too high or too low for the computer's internal audio circuits to play properly. The result will be alias distortion and other sonic weirdness. This will not affect your waveform data, it just sounds strange and is not representative of the true sound which may be properly played back by a dedicated sample module or keyboard. Also, the mouse response at very high and low pitches may be somewhat sluggish, in that moving to a new key or releasing the mouse button will not change or stop the sound instantly.

Scrub Mode

There is a fourth Mode choice for the editor windows, and that's **Scrub**. This is the last item under the Mode menu, joining the Normal, Freehand Draw, and Zoom Box modes. Scrub allows you to hear a sound by moving the mouse horizontally across it. This is useful when you need to zero in on a particular fragment of a sound, perhaps for an edit. It gets its name from the process of manually rocking the tape reels of recording decks in order to find precise splice points. The tape was said to scrub across the playback head. Scrub mode can be used in either the main or overview portions of an editor window, but for best use, the range of interest should be narrowed down to perhaps 3 to 10 seconds. (This is relatively easy to do when using the **Show Position** option under Playback Preferences.)

When you select Scrub mode the mouse pointer turns into a pointing finger. To use scrub, move the mouse to an area of the wave which you'd like to listen to. Once there, depress and hold the left mouse button and move the mouse along the wave. When you reach the end of the area of interest, release the mouse button. The wave will start to play back. The playback direction will depend on which direction the mouse was traveling. Moving the mouse left to right causes ordinary forward playback while moving it from right to left will cause the sound to play in reverse.

Sometimes the visual depiction of a sound can lead you astray. Most people expect to see a string of words represented as individual blocks of sound, but this is not always so. Many times words run into each other creating a single visual unit. On the other hand, some words may come out in two or more distinct parts. For example, a word such as a "boat" will very often appear as two separate chunks; the main portion being the beginning B and vowel sounds, and a second portion representing the ending consonant T. These segments can be delineated via Scrub.

Real-Time Interactive Preview

Many signal processing functions and effects dialog boxes allow you to preview their results. This can be very handy since you might want to control the settings "by ear", and do not wish to continually undo edits that were not quite right. This preview process is immediate- you do not have to wait for Wrench to create temporary playback files or things of that nature. To use Real-Time Preview, adjust the controls on the dialog box to your desired settings and then hit the Preview button. Playback of the edit affect area will begin in a moment, with your control settings in use. The section will play through once. If you wish to terminate playback early, simply hit the Preview button a second time. In many cases, the playback is interactive. In other words, it is updated as you change the parameters in the dialog box. For example, if you are experimenting with vibrato using the FM effect, you can move the Mod Speed slider and hear what it does while the sound is playing. Details on which parameters are available for interactive updating can be found in the dialog box descriptions (when in doubt, hit the Help button in the dialog box).

Functions and effects with Real-Time Preview include: AM, Chorus, Combine Samples, Compressor, Cross Multiply, Echo, Envelope Generator, EQ, Flange, FM, Grunge, Noise Gate, and Transfer Function. Some functions require more computing power than others during playback. If the process is too intense for the computer to keep up with, you will hear gaps and clicks during playback. If this happens, increase the playback buffer size using Playback Prefs (found under the File menu). This problem is exaggerated at higher sampling rates and for stereo sounds. For faster computers, smaller buffers are generally preferred in order to keep interactive updating quick and responsive. Finally, if your sound card uses only a limited number of playback rates, you may hear pitch shifts or alias artifacts during preview.

Recording Waves

Besides being used to edit existing material, Wrench can be used to create brand new sounds. There are two ways to do this: you can either record a sound using your sound card's recording capabilities, or you can generate a sound mathematically. Let's take a look at recording.

Before you record something for the first time, make sure that you have read the materials which came with your sound card. Many possible initial problems can be avoided by simply taking the time to properly identify all connectors, switches, and settings. Before recording, use **File/Record Prefs** to set the basic operation of Wrench's record function.

Under **Record Prefs** you can set whether you want **mono** or **stereo** sounds, the **record buffer size**, the bit **resolution**, the **sample rate**, and the **driver**. The buffer size can be set to Small or Large. Small requires less system resources and produces the snappiest VU meter response while recording, but if you notice gaps and clicks in the resulting sound file, switch to Large. **High Res 24 Bit** is useful if you have a sound card capable of greater than 16 bit resolution. Selecting this will use the high resolution recording capabilities instead of standard 16 bit CD quality mode. The resulting sound file will be saved using 32 bit floating point values and will be approximately twice as large as ordinary 16 bit WAV files. High resolution mode is often more taxing on the computer system and should be avoided if your computer is producing skips or gaps while recording. If High Res is selected but your sound card does not have high resolution capabilities, you will be warned about this when you try to record something.

The Sample Rate selection includes the standard rates of 11.025 kHz, 22.05 kHz, and 44.1 kHz (CD standard). You can also select you own custom rate. Please be aware that not all sound cards are capable of custom recording rates. Custom rates are handy if you need to match rates with some other device. For example, you might have a 12 bit MIDI sampler which records at 32 kHz. A good 16 bit sound card may offer improved fidelity, so you can use the sound card to capture the sound at the same 32 kHz rate (thus avoiding Sample Rate Transpose). The available drivers are presented in a list box. Simply choose the one you want. You can also select from the Presets list for common arrangements. You can save and recall your own custom settings via the Save Preset and Load Preset buttons. All of your preferences stay in force until the next time you call up Record Prefs.

Once the preferences are set, you can start recording. File/Record will first prompt you for a file name for the new sound. It will then bring up the **Record Dialog**. This works much like an ordinary tape recorder in that you can record and re-record sections, and play parts back. Along the top are a group of buttons. The most important buttons in this group are **Record** and **Stop**. As you would expect, Record starts the recording while Stop ends it. To move around in the sound file, use the **Goto buttons**. Goto Start "rewinds" to the very beginning while Goto End "fast forwards" to the very end of the sound file. The Goto Offset allows you to jump to any location within the sound file. The **Play button** will start playing the sound at its present location. (eg, if you rewind to the start, playback begins from the start of the sound file). Recording is not a one shot, do or die process. Like a normal tape recorder, you can move to new locations and re-record sections. Also, if you're at the very end, you can continue the recording.

To help with level settings, you will find **four meters** in the middle of the dialog. The left channel meters are at the top with the right channel meters below them. The innermost meters are normal continuously responding meters. The outer meters are peak-hold type, meaning that they always show the highest signal encountered so far. The peak-hold meters may be reset to minimum by hitting the **Reset Peak** button. Wrench's meters are true peak responding in that even very short transients will be properly represented. If the meters never go into the red, you can be assured that the recorded wave is unclipped. You can monitor signal levels without recording anything by hitting the **Monitor** button.

Below the meters is a readout of your settings including the present file position, size, and name, the number of channels, bit resolution, and sampling rate. Wrench always records sounds to disk using the 16 bit WAV format. The **Auto-Load** check box allows you to have the disk file automatically loaded into the editor window when you are done recording (useful if further editing needs to be done at that time).

Generating Waves

In contrast to directly recording sound sources is **Generate**. This function allows you to create simple waveforms from scratch. There are several uses for this including test tones, base waveforms for AM, FM or additive synthesis, and as material for the Cross Multiply function. The first thing you have to decide on is a waveshape. Your choices are: **Sine**, **Square (true)**, **Triangle (true)**, **Sawtooth**, **Variable Pulse**, **Noise (white)**, and **Silence**. The **sine** wave is the simplest of all waves and contains no harmonics. The **square** and **triangle** waves are referred to as "true" since they are properly bandwidth limited and contain no alias components (ie, for you mathematical types, they are built from a Nyquist limited Fourier series). The **sawtooth** and **variable pulse** waveforms are not bandwidth limited. The variable pulse waveform is a rectangular wave where you have control over the duty cycle (ie, the "high" time versus the entire cycle time). All of these waves contain a good deal of harmonics and are useful for synthesis. A **white noise** source is also available which can be useful for the synthesizing percussion instruments. The final waveform isn't a waveform at all, just **silence**. Besides being handy when you need a gap or pause, this is particularly useful for creating your own waveforms in conjunction with the Enable scripting language (see the Enable section for details and examples).

Once the waveshape is decided, you need to specify the **sample rate** of the waveform and its **fundamental frequency** (pitch). Some sources such as noise or silence do not have a pitch, so the Frequency value is ignored. Finally, you need to specify the **duration** or length of the new sound. For starters, you may wish to scan the Presets list to see if what you need is already available. You can save or recall your own custom settings via the Save Preset and Load Preset buttons.

Disk Files

Waves may be stored on disk for future use and archiving. Several formats are available, with AIFF and WAV being the more popular choices for 16 bit material.

To save a wave to disk, select **File/Save** from the editor's menu. This will use the present path (disk, directories, and file name). If the present path has not been set (ie, this wave was loaded via MIDI), the file dialog will pop up allowing you to set the disk, directories, and file name. Finish the save by selecting the OK button. If you wish to save under a new path or name, select **File/Save As**. This will also bring up the file dialog, and you will have the option of overriding the default format type, if desired.

To load a disk based sound, select **File/Open**. The standard file dialog will pop up allowing you to easily find the wave on disk. After selecting the desired file (with the mouse or keyboard), select OK to load the wave. Once the wave is loaded, it will be drawn into the editor's window at full scale. If a wave already exists in the editor, you will be warned that loading will destroy the present wave (you may abort the load if desired).

In order to speed redrawing of large sounds, Sample Wrench maintains special graphics information about the waveform. This information will be automatically saved to disk when you save the sound file itself. This graphics file will have the same name as the sound file, but Sample Wrench will add the extension **.dgc**. When loading a sound, if a companion .dgc file is found, it will be used, which helps to speed the loading of sound files. Do not attempt to load the .dgc graphics file itself.

And now, some details on the different file formats available:

WAVE
Also called .WAV, these files generally come in either 8 or 16 bit versions, some with compression. Wrench can read and write 8, 16, and 24 bit integer, and 32 bit floating point forms in either stereo or mono using standard uncompressed PCM, and also read 8 bit u-law or a-law compressed forms. Sample Wrench will save and load information on keymapping, finetuning, markers, loops, and sample name. Markers and loops are limited to IDs less than 256. Saving this extra information is optional since some programs cannot properly read WAV files unless they are of the simplest form. To enable saving of this info, check off the desired parts under **Format/WAV Options**. These chunks are smpl (loops, root note, and fine tune), inst (full keymap and fine tune), cue (markers), and INAM (sample name). Note that Wrench is intelligent about saving chunks. For example, if your wave doesn't have any markers, then a cue chunk is not saved (even if cue is checked off).

8SVX
Originally created for the Amiga, this is an 8 bit mono format. 8SVX is popular for space-sensitive applications where high fidelity is not the primary concern.

AIFF
Wrench can import and export stereo as well as mono AIFF samples in either 16 or 24 bit styles. Also, Wrench can load samples outside of the 16/24 bit formats. Samples with less than 16 bit resolution are padded out to 16 bits. There are some important points to remember when loading/saving AIFF format sounds. First, AIFF only allows two loops (sustain and release). Wrench allows up to 256 loops, but only the sustain and release loops can be saved with the AIFF format. Similarly, Wrench allows 256 markers (0 through 255), whereas AIFF allows 65535 markers (1 through 65535). When saving, markers 1 through

250 are saved "as is". Marker 0 is saved as marker 251 since there's no such thing as "marker 0" in AIFF. Markers 252 through 255 will be used to save the sustain begin/end and release begin/end since AIFF defines loops with markers. This also means that when loading AIFF files all markers greater than 255 will be ignored by Wrench. As a convenience, once Wrench has created the sustain and release loops, it will delete the associated AIFF markers for you.

AU
This is the format used by Sun workstations. It is also used by Next computers, although they use a .snd extension. There are many variants of au files. The more common form is mono 8 bit u-law companded. This is the form Wrench uses for saves. For loading au files, Wrench also recognizes 8 bit and 16 bit linear PCM along with 8 bit a-law companded.

RealAudio
This is a series of formats designed by Progressive Networks, Inc. The main idea idea is to compress the audio so that it can be transmitted in real-time over a network. Normally, these files have a .ra extension. Although Sample Wrench can encode (write) RealAudio, it cannot decode (read) RealAudio. If you save a file using this general format, a dialog box will pop up allowing you to select the desired compression method and set other file attributes.

SND
Unfortunately, there is no single standard for files with a .snd extension. The two most popular uses of .snd are on the Next and the Macintosh. The Next version is essentially the same as the Sun AU format, so use the au choice for this case. On the Mac, .snd files are generally 8 bit raw files. Use the RAW 8 unsigned choice for this case.

Sound Designer Type 1
This is a 16 bit mono format with some marker and loop capabilities. It is popular on the Macintosh.

Studio 16 Version 3
This is a 16 bit mono format used on the Amiga. All files exported from Studio 16 must have their edits "made permanent". There is no marker or loop capability when using this format.

VOC
This extension originated with the Creative Labs Sound Blaster series of audio cards. Wrench reads 8 and 16 bit linear PCM forms (VOC version 1.20 or higher), and writes 8 bit linear PCM.

RAW
These are very simple files in that they contain only sample data, and nothing about markers, loops, names, or even sample rates! There are many programs on various platforms which utilize simple raw files, and Wrench users have access to them too. In fact, with proper use of the raw formats, you can read almost any uncompressed mono sound file into Wrench.

In spite of popular myth, there is no universal raw format. The interpretation of the data depends on the type and source platform. Generally, the data type is linear PCM using either 8 bits or 16 bits. 8 bit data is platform independant since all computers store 8 bit quantities the same way. The only question remaining is whether the data is signed or unsigned (ie, is set up as all positive values, or as a mixture of positive and negative values.) Many 8 bit sources use unsigned data, but Wrench has a choice for either 8 bit signed or 8 bit unsigned. In the 16 bit world, almost all sources use signed data so this is not a problem. On the other hand, 16 bit data may be stored in "ordinary" or "byte-reversed" forms. This depends on the microprocessor used in the system. Motorola 680X0 platforms such as the Amiga and Macintosh are opposite of the Intel 80X86 platforms (clones). To compensate for this, Wrench has choices for 16 bit "Motorola" and 16 bit "Intel". Finally, Wrench has choices for 8 bit a-law and u-law companded files. u-law (the u is shorthand for the Greek letter mu) is the telephony standard used in the US, while the similar a-law standard is used in

Europe. Companding is a technique which tries to create constant percentage resolution which is tied to signal amplitude. This generally enhances the dynamic range and noise performance for sources such as human voice.

Since there are no "tags" inside of a raw file (only sample data), Wrench cannot intelligently scan the file and figure out which form is being used, as it can with other formats. Wrench has to know which raw format you want before you load the file so a Raw Open dialog will pop up, asking you to choose the import type. Since this dialog can be a pain to folks that are making batch scripts with Enable, the situation is handled a little differently. In this case, Wrench will examine the File Format menu and see which item has been checked off. It will use this to load the file. (Remember, this is true only for raw format files, other types can be intelligently scanned by Wrench and the menu item is only used to indicate the desired save format). If you do not check a raw file type and Wrench cannot determine what the file type is, it will load it as an unsigned 8 bit wave.

Since Wrench can always load a file as raw (memory and file integrity permitting), you can use this facility to load almost anything into Wrench. If, for example, you have a collection of sounds which use a proprietary file format based on 8 bit unsigned data, you can probably pull them into Wrench with little problem. Usually, file formats consist of a short "header" which contains information on the sampling rate, markers, and the like. After this comes the sound data. If you load this in using the unsigned 8 bit raw selection, the entire file will be read in as sound sample data, including the header. You'll be able to see (and "hear") the header in Wrench, at which point you can simply clip it out, leaving you with the waveform data. For 16 bit data the issue includes a minor twist. The header might be an odd or an even number of bytes, and this can misalign the sound data. If Wrench senses that this could be a problem, it will ask if you'd like to skip over a byte in order to align things. If you don't know, simply say Yes, and if the resulting sound is nothing but a distorted mash, try reloading and saying No. For the very adventurous experimenter (and those who simply like to do weird things), you can load in non sound files, such as text files, pictures, or even other programs. Normally, you'll get a lot of static and noise, but every now and then you can come up with something interesting. As far as proprietary sound files go, the raw readers will not normally give good results on files that use compression or which interleave data (some stereo types do this). Finally, please note that raw files do not have a name, markers, loops, or a keymap. The default sampling rate is 44.1 kHz for 16 bit files, 22 kHz for 8 bit files, and 8 kHz for u-law and a-law files (you can, of course, change these inside Wrench using Sample Rate Transpose-Rate Only).

View Options

There are a number of ways in which you may configure a given editor. These configurations do not change the wave data, they simply let you view it in different ways. Appropriately enough, these items are all found under the **View** menu for the editors. Each of the editors may be configured differently. These items are attributes, and as such, use a check mark to show when they are active.

First of all, you have a choice of units for the horizontal and vertical axes. The horizontal axis has calibration points at the very start and end of the displayed portion. The vertical axis shows points at the top, bottom and middle of the displayed portion. The horizontal axis may be calibrated to read time, sample number, measures, beats, or frames. These choices are selected via the **View/Horizontal** menu item. The default is set for total seconds. If you select a time format, the value can be shown in terms of total seconds, or in minutes:seconds format. In either case, very small values are automatically scaled back to milliseconds for you (1/1000ths of a second). While the use of a time scale may be most natural, sample number calibration is very useful when using markers, loops and certain functions.

Units of measures or beats are useful if you're working on sampled sections which are part of a longer song, and you need to coordinate your editing with your sequencing. The default settings are a tempo of 120 beats per minute and 4 beats per measure. If you select units of Beats, the horizontal axis will be shown in beats, based on your tempo. If you choose Measures+Beats, the axis will appear in this form: MM+BB, where MM is the number of measures elapsed (based on your tempo and the number of beats per measure), and BB is the number of beats within the given measure. For example, If you have set beats per measure to 4 and the sample is exactly 10 and one half measures long, the Beats choice will show a maximum of 42, while Measures+Beats will read a maximum of 10+2 (ie, one half measure is 2 beats).

Frames per second modes are useful if the waveform must be linked to video or movie scenes. You may choose between 24, 25, 30, and 30 drop frame modes. You also have the choice of displaying the time as total frames or in hours:minutes:seconds:frames format.

The vertical axis may be set to read in sample value, percentage of full scale, and positive or negative decibels via the **View/Vertical** menu item. Since a 16 bit sample can contain over 64,000 discrete values, the sample value scale will run from over +32,000 to -32,000 at maximum view. This is true for Wrench 24/96 as well, since the final result will most likely be targeted to 16 bit audio CDs. The percentage scale runs from +100% to -100% at maximum view. The decibel scale has three variations: dB, -dB, and -dB RMS. The straight dB form runs from 0 dB at the origin to greater than 90 dB at the maximum, while the negative dB scale sets the maximum level at 0 dB with the origin being -96 dB. The -dB RMS choice is similar to -dB in that the maximum obtainable level is 0 dB and the minimum is -96 dB. Unlike the other vertical forms, the vertical position of the mouse does not affect the XY Readout Y value (more on XY Readout in a moment). The Y value does not represent the vertical value of the place where the mouse is pointing. Instead, this form reports the RMS value of the sound sample in the vicinity of where the mouse is pointing in time (along the horizontal axis). The RMS value gives you an indication of the relative loudness of the sound sample at that point. Note that as the mouse is swept from left to right with an ordinary decaying sample, the Y value will change; however, if the mouse is kept at the same horizontal position and only moved up and down, the Y value will not change. In essence, the Y value is giving you a numeric readout of the envelope of the sound, an idea of how its volume changes over time. Its primary use is as an aid in helping you to set proper Threshold levels for the Compressor function, but it can be used for other purposes. Due to the calculations required to compute an RMS value for the sound (instead of the instantaneous level associated with the other forms), you may notice that the XY Readout update rate is somewhat slower.

Decibels are naturally a "magnitude only" measurement, and thus, positive and negative signal levels of the same intensity will give the same decibel reading. These choices are selected via the View/Vertical menu item. The default is set for percentage. Please note that you are not locked into a given calibration, you may change it whenever you like.

The next item under the Options menu is **Box Outline**. Selecting this draws a box around the wave display area. Its use is purely aesthetic. The default setting is box drawn.

One nice feature for use at high magnification is **Color Point**. To get this feature, select **View/Color Point**. When active, each sample point will be highlighted in a complimentary color. This makes them very easy to spot. It is particularly useful when performing very accurate editing such as marker placement or free hand drawing.

Overview is the next attribute under the Options menu. Once you have zoomed into a wave, you may find it difficult to remember exactly where your viewing position is relative to the entire waveform. When selected, Overview draws a small display of the entire wave above the normal working display. The area which is presently in view (ie, the portion you have zoomed into) is shown on the overview in reverse highlight. Please note that the XY Readout function, the Zoom Box, and Marker/Loop grabbing all operate in the overview area as well as in the main area. Also, when Overview is selected, the edit Affect area will be indicated on the Overview as a highlighted line across its top edge.

Many times, it is useful to know the horizontal/vertical values associated with a specific spot on the wave.

While any value may be estimated by using the calibrated axes, selecting **View/XY Readout** makes things very easy. When selected, the XY co-ordinates of the area under the mouse pointer will be printed in the upper left corner of the editor's window. All you have to do is point at the spot you're interested in. As you move the mouse, the values will change accordingly. This can be very useful when performing cut and paste operations, or when using the mouse to position markers. The default setting has this feature enabled.

The colors used for the waveform drawing are defined through the **Set Colors** item. You have control over the colors used for the waveform itself, background, markers, loops, axis, labels, and color point. This dialog presents a set of Red, Green, and Blue color sliders which are used to mix a specific color for a given drawing component. First, select the component you'd like to alter, and then adjust the Red, Green, Blue sliders to your preference. A Default button is available to set the colors back to the Wrench startup values. Also, you can choose to reset all open editors to this new color scheme by selecting the Apply To All Editors check box.

The **Set Font** item allows you to specify the style and size of the font used in the editor windows. This uses the system standard font selection dialog. In general it is best to use simple fonts of modest size. If the font is too large for the editor window, then nothing can be drawn.

Associated with the horizontal axis is the **View/Set Offset-Tempo** item. This allows you to add an offset to the displayed wave. For example, you may be editing a 5 second sound effect which begins in a film score at 43 minutes, 11 seconds, 18 frames. You can add this time as an offset to the displayed horizontal units, so that you can easily keep track of where you are relative to the film score. Set Offset can be used as seconds/frames/sample number calculator, and it is also used to define the Beats Per Measure and Beats Per Minute values if you're using the beats or measures axis calibration. To use the calculator feature, type the desired value into the appropriate slot (such as Total Seconds), and then hit the Calc button next to it. In a moment all of the other slots will be updated to show equivalent values.

The final set of items in the Options menu deal with **Alternate Views**. Each editor has available up to ten Alternate Views. A View reflects the portion of a wave you are looking at. In other words, this represents the horizontal and vertical magnification levels, and the horizontal and vertical offsets. You might set one view to be a close-up of the attack portion of a wave, and another to be the very end of a wave. You can think of selecting a view as a sort of zoom in-out with scroll macro. If you need to compare various sections of a wave quickly, the Alternate View feature is very handy. Possibilities include examination of critical areas, comparing potential splice points, and checking the start and end of loops.

The views are defined using the **Set View** menu item under the View menu. Selecting a view number defines that view to be equal to the editor settings presently chosen. Thus, to define view number one to be a close-up of the attack portion of a wave, first zoom into the starting area, and then scroll until you have the exact portion of the wave in which you're interested showing in the editor. Now select View/Set View/1. View number one is now defined. You can now move to any other part of the wave, zoom in or out, or even do a Show Full. To recall a view, use **Get View**. In this example, selecting View/Get View/1 will automatically adjust things so that you're looking at the attack portion of the wave again, no matter what your present view happens to be.

Direct access to Views is possible via the Views floating toolbar window (under **Setup/Toolbars**).

Specifying Markers

Each virtual editor may have up to **256 markers** associated with its wave. A given marker signifies a specific sample in a wave, rather like a book mark. These markers can be used for a variety of purposes. They can be used to specify clip boundaries, edit ranges, or other edit points. You may also use them to keep track of points for your own use. Markers may be freely moved around a wave at will. You have the option of displaying markers directly on the edit window, or making them invisible (for a less cluttered display). When displayed, markers are drawn as a thin vertical line at the specified sample position. Each marker will have its ID number written along with it. Even numbered markers have their number placed at the base of the line, while odd numbered ones have their value written at the top of the line. To make markers visible, select **Loops+Markers/Marker Show** (ie, make it checked). You can grab and move markers with the mouse. To do so, position the mouse slightly above or below the ID number (ie, toward the middle of the waveform display) and depress the left mouse button. A highlight position bar will appear and track your mouse movements. Position the mouse where you'd like the marker to be and release the mouse button. The display will be redrawn indicating the new marker position.

To set a marker to a desired sample, select **Loops+Markers/Marker Set**. This will bring up the Set Marker dialog. In order to select the marker you wish to edit, enter its ID number in the ID text box. The display will be updated to show the values for this marker. If you prefer, you can type the name of the marker into the Name slot for the same result. If this ID or Name cannot be found, Wrench assumes that you wish to create a new marker, so it displays a checkmark in the New check box. (If you want a new marker, just click on the New check box and Wrench will make an appropriate ID for you). The Marker Set dialog includes a set of arrows so that you can easily scan the list of existing markers. (New markers are placed at the beginning of the list.)

Markers may be positioned by typing in a sample number or by using the mouse to point at a sample. For the first form, just type the desired number into the **Position** text box and then select the **Value** button. For the second form, select the **Mouse** pushbutton. This automatically closes the dialog and turns the mouse pointer into the I-beam insertion/position pointer (a vertical line with handles on it). To use this, move to the desired section of the wave and press and hold the left mouse button. To help you see, a vertical line is drawn at the mouse position in a complimentary color. As long as you hold down the mouse button, this line will follow the mouse pointer. When the button is released, the line turns into the marker. If you need some adjustment of the view, the toolbar buttons and border sliders remain operational for you. If you suddenly decide that you would like to abort the marker placement, just click outside of the active area (for example, between the left window border and the vertical axis). You can also delete a given Marker, or all Markers from the Marker Set dialog by selecting **Delete** or **Delete All**, respectively.

Dropping Markers On-The-Fly

Sometimes you may want to create a bunch of Markers by simply pointing and clicking at the waveform and you don't really care about naming them or being sample-accurate. You can do this by holding down the SHIFT key and then pointing and clicking on the waveform. Wrench will create a new Marker for you at this spot. These Markers will be numbered from 0 on up, skipping over any existing Marker IDs. The Markers won't have names, but you can always go back and give them names using the Set Marker dialog. Thus, by holding down the SHIFT key, you can create Markers as fast as you can point and click, and never have to open the Set Marker dialog. Note that you can drop Markers during playback, which can be very handy, especially when used with the Show Position playback bar. This capability is also very handy when using the Cut/Keep List function.

Specifying Loops

Loops are used to effectively lengthen sounds. Some sounds can sustain for very long periods of time. It is often not practical to record the entire event, so sections of the sound are played back repeatedly. These sections are referred to as loops. If done correctly, loops can be very effective. Virtually all modern samplers allow some form of looping. Some units use a single loop, and others allow for multiple loops. Sample Wrench let's you create upto 256 loops, including a sustain loop and a release loop. Historically, sustain loops were used to hold a sound while a key was depressed. In contrast, the release loop was used for the fade-out (when the key had lifted). These terms are hold overs from synthesizer days and their use of amplifiers to create an ADSR (attack, decay, sustain, release) envelope. Two different forms of looping are also supported. Loops may be either Forward Only or Forward Backward. A Forward Only loop works like a tape loop; once the end point is reached, the next sound heard is the very start of the loop. A Forward Backward loop plays normally until it hits the loop end. Once there, the sound is played backwards from end to start. In other words, the sound bounces back and forth between the end points. Note: many samplers do not support Forward Backward looping.

In order to define a loop, start and end points are required. To set a loop, select **Loops+Markers/Loop Set**. The Loop Set dialog has slots for the loop ID, IDs for the Sustain and Release loops, start and end offsets, loop type, auto-location, and whether or not the loop will be displayed in the editor window. The Start and End slots are used to specify the loop points, the Auto Locate check box indicates whether auto-location is in force, and the New check box is used if you wish to create a new loop. As you can see, this dialog is similar to the Marker Set dialog in operation. The Type buttons indicate whether this is a Forward or Backward type loop. Finally, a set of arrows allows you to scan the list of existing loops.

To edit an existing loop, enter its ID number into the ID slot. The display will be updated with data for this loop. If this ID does not exist, the New check box will become checked. Also, note that the IDs for the sustain and release loops will appear in their respective text boxes. To change the loop which is to be used as the sustain or release loop, type its ID into the proper text box. If you don't want a sustain or release loop, use an ID of -1 for them.

To create a new loop, click on the New check box. A new loop will be created and inserted at the beginning of the list. Set the parameters as needed, including loop start and end points, loop type, and whether or not you'd like this loop to be hidden from view (if Show is not checked, then the loop will not appear in the editor window when Loop Show is enabled. This allows you to make a less cluttered display). When you are done, select **Value** to keep the loop, or Cancel to pretend that you never opened this dialog in the first place. As its name suggests, the **Delete** button is used to remove a loop from the list. **Delete All** will remove the entire loop list.

Another option is to use the button labeled **Mouse**. This button allows you to define both a loop start and end in a single move. If you select it, the Loop Set dialog will disappear and the mouse pointer will turn into the Insert pointer. You can now draw where you'd like the loop to be. Pressing the left mouse button starts the process and releasing it ends the process. The area between the loop start and end will be drawn in complement mode so that it is easy to see. The whole affair works pretty much the same way as defining a clip with the mouse (next section).

If Auto Locate is active, the loop start and end points will be verified as per the auto-location requirements. As with Marker Set, you can jump to the next auto-location by simply increasing the value given by one. For fine adjustments to the loop points, close the dialog and either move them directly with the mouse (as outlined earlier) or use the Loop Window function. Also, note that it is not uncommon to have only a single sustain loop and completely ignore the release and other loops. For example, many samplers only take into account a sustain loop, so why bother with anything else?

You can get a visual indication of your loops by selecting **Loops+Markers/Show Loop**. If you select Sus/Rel Only, the sustain loop is drawn at the bottom of the wave and the release loop is drawn at the top.

An easy way to double check marker positioning and loop settings is to select File/Info. Both sustain and release loop start and end points are displayed. Like markers, you can grab and move loop end points with the mouse. The grab areas, so to speak, are the areas next to the loop's IDs.

By the way, Wrench automatically adjusts loops and markers if you perform any operation which will make the wave smaller (such as cutting a clip). Markers or loops which wind up being beyond the end of the wave are deleted. If only the end of a loop is past the end of the wave, the loop will be kept, but the loop endpoint will be truncated to the end of the wave. Also, If you use Overviews, note that you can grab markers and loop points there. Thus, it is possible to position markers and loops independently from the present main view.

Finally, please remember that for playback, Wrench only looks at the sustain loop (or lacking that, the release loop). Virtually all modern samplers use this same scheme. The ability to create more than two loops in Wrench is primarily for convenience during comparisons (ie, all you have to do is reassign the sustain loop ID number instead of resetting and remembering start and end positions).

Auto-Location for Loops and Markers

Placement of markers and loop points can sometimes be a tedious process, particularly if there are certain constraints on where it should be placed. To help alleviate this, Wrench features **Auto-Location** of markers and loops. This is kind of like the "snap to grid" ability found on certain CAD and DTP programs. Markers and loops can be roughly positioned, and will then be automatically repositioned to the next sample which meets the set requirements. To set the auto-location parameters, select **Loops+Markers/Auto-Locate**. This dialog lets you specify a multiplier, an offset, whether or not match level detection is used, and the minimum and maximum values for the match level. You can also set requirements for the waveform slope. The multiplier makes sure that only sample numbers which are an integer multiple are acceptable. For example, if the multiplier is set to 3, only every third sample will be allowed. This can be very useful when setting loops for certain samplers. For example, a given sampler may require that loop points lie on a page boundary (ie, a multiple of 256). The offset specifies how much of the start of the wave is to be ignored. If offset is set for 2000, only samples from position 2001 and on are acceptable. This is quite useful for skipping over attack transients. Match Level detection specifies that a sample is valid only if the data value at that point is between the minimum and maximum match levels. The level may range from approximately -32768 to +32767 for 16 bit data Wrench. For Wrench 24/96 with 32 bit data, the range is in percent from -100 to +100. Match level detection can be very useful when trying to set seamless loops. A helpful hint: If you are looking for "zero crossings", set the minimum and maximum match levels to 0. Note though that with 16 bit waves, exact zero crossings are not required for good loops (and the average 16 bit wave will have very few of them anyway). You may find a range of -.1 to +.1 to be more practical (about +/-32 for 16 bit data Wrench). For that matter, good loops may be obtained at identical peaks (this is very true for backwards-forwards type loops, if your sampler supports them). In cases like this, you might set the range at something like +80 to +81 (or 26200 to 26500 for 16 bit data Wrench). These values are highly dependent on the wave, of course. The Slope group allows you to specify the direction in which the wave is headed (Positive or up, Negative or down, and Zero or level). Finally, these four items may be combined. It is possible to specify an offset and match level together, or some other combo as well.

Once the auto-location parameters are set, they become active by selecting the **Auto Locate check box** in the **Marker Set** and **Loop Set dialogs.** Auto Locate may be used with either mouse or keyboard marker/loop positioning. When using the Position text box with Auto, type in the desired value. Wrench will calculate the next appropriate sample number and print it out in the Position text box for you. To get the next value, simply add one to the value given. When using the mouse, Wrench will find the next appropriate sample after your placement. Note: it is quite possible that there will be no more "appropriate" samples in your wave, so watch for this (setting Loops+Markers/Marker Show and Loops+Markers/Loop Show helps a great deal)!

System Clipboard

Sample Wrench also offers access to the Windows system clipboard which allows for easy cut and paste operations, and the ability to quickly move sound clips between other applications which support the clipboard. The system clipboard is not as extensive as Sample Wrench's internal clipboard. Most notably, only one clip can be stored in it at a time. If you are familiar with the cut and paste operations found in most word processors, then the following operations should be almost second nature.

The **Copy** menu item copies the edit Affect area to the system clipboard. **Cut** is similar, but also removes the edit Affect area from the wave. **Paste** removes the edit Affect area and replaces it with whatever is in the clipboard. This sequence is particularly handy to use if you set the edit Affect type to Affect Mouse. In this way, you can quickly define areas with the mouse and then cut, copy, or paste them into any of the open editors. When used in conjunction with the Mouse Shuffle feature (see the prior section under ~~Edit Modes~~), cutting up and re-arranging blocks of audio is very fast and convenient. For a simple insertion (i.e., no area replaced when using Paste), set the Affect area to a single point. When using Affect Mouse, this is as simple as just clicking on the wave where you'd like to insert.

Using Multi-Clips

The Multi-Clipboard is Wrench's internal clipboard. It can hold a large number of audio clips, each with their own name. It is much more powerful than the standard system clipboard, but somewhat more difficult to use if all you want is a simple cut or paste. For that, we recommend that you stick with the system clipboard (next page).

All Multi-Clip functions are accessed via the **Edit/Multi-Clips** menu for the editor. In essence, clips are wave fragments. You can think of them as temporary storage slots. You can use clips to remove a section of wave, re-arrange a wave, or trade portions of waves between the virtual editors. Access to clips is done through the clipboard. Sample Wrench allows you to have as many clips stored in the clipboard as you want (limited only by your system memory). To help you keep track of things, each clip may be named. Clips are defined through the use of either markers or the mouse. Each editor has an active clip associated with it. By default, the **active clip** is the one most recently created by that editor (this can be changed, indeed, the active clip may be null - in other words, no clip at all). The active clip is the one which will be **Cut**, **Copied**, **Pasted**, **Replaced**, or **Erased**. Functionally, the clip routines are quite like the text-block capabilities found in many word processors.

To create a clip, select **Multi-Clips/Clip**. The Clip dialog will always bring up the lowest numbered empty slot. This number is found directly below the dialog's title. next to it is a text box which allows you to name the clip (you don't have to if you don't want to). The clip may be defined by either specifying a pair of markers, or by using the mouse. To use markers, first set the marker values using Loops+Markers/Marker Set. Once the markers are set, call up the Clip dialog, type the marker numbers into the Start and End text boxes, and select the **Marker** pushbutton. To use the mouse, simply select the **Mouse** pushbutton. The mouse pointer will turn into the insertion/placement pointer. Move the pointer to one end of the desired clip area and press and hold the left mouse button. While still holding the mouse button, drag the pointer over the desired clip area. You will note that the encompassed area will be shown in reverse highlight. When you get the clip the way you want it, release the left mouse button. You can also use the presently defined edit Affect area to define a clip by selecting the **Affect** button.

This freshly made clip is now the active clip. It is very important to note that the clip is only referenced at this point, it has not been copied. The reason for this is that clips can be very large and thus consume a lot of memory. This would be very inefficient if you just wanted to cut the clip out of the wave. As long as the original wave remains unaltered, this clip can be freely used (for example, transferred to another editor). As soon as the original wave is altered, the clip is discarded (its reference has changed, and is thus no longer valid). If you want to hold on to the clip, you must Copy it. To do this, select **Multi-Clips/Copy**. Once a clip has been copied, you can do anything you want to the original (including deletion) and the clip will remain untouched. If you would like to remove the clip section from the wave, select **Multi-Clips/Cut**. Cut is a lot like a tape edit, but with microsecond accuracy. Of course, a clip can only be cut once! You may also wish to insert the clip into the wave at another point. To do this, select **Multi-Clips/Paste**. The Paste dialog lets you specify the insertion point by using a marker or the mouse. To use a marker, type the marker number into the text box (remember, the marker must be specified via Loops+Markers/Marker Set first). To use the mouse, select the mouse pushbutton. The mouse pointer will turn into the insertion/placement pointer. Move the pointer to the desired area and press and hold the left mouse button. Doing so will create a reverse highlight guide line. As long as the mouse button is held down, you can sweep around the wave until you get to the exact position you need. Once in position, release the mouse button.

Replace is very similar to Paste except that it doesn't "make room" for the clip by sliding the second portion of the wave data toward the back of the wave. Instead, Replace overwrites the existing wave data.

After you have finished with a particular clip, it is wise to erase it. This frees up valuable memory. To do this, select **Multi-Clips/Erase**. Remember, this only dumps the active clip for this editor. If you want to dump all of the clips, select **Multi-Clips/Erase All**.

In order to change the active clip, select **Multi-Clips/Edit**. This brings up the Clipboard dialog. Each of the available clips are shown. The clip's ID will be shown first, followed by its name and size in sample points. If the clip has been copied, a "C" will appear next to the ID, and if it's a stereo clip (ie, clipped from a stereo wave) an "S" will appear as well. At the bottom of this list is a line telling you which clip was the initial active clip for this editor. To change it, just click on the clip you'd like (ie, on its number, name, or size) and select OK. It is also possible to delete clips from this dialog. To do this, just select the desired clip from the list and then select **Delete**. This is a lot quicker than repeatedly selecting new active clips and then going back to the Clips/Erase menu item. Clips may be previewed from this dialog by selecting the desired clip and then hitting the **Play button**. If you would like to save a clip directly to disk using the present Format choice, select the clip and then hit the **Save** button. In like fashion, wave files may de directly imported into the clipboard by using the **Load** button.

The final item is **Multi-Clips/Play**. Selecting this will allow you to hear the active clip.

Time for an example. Let's say that your wave contains the spoken words "I love Karen". Words are generally rather easy to pick out, so it should be no problem to create a clip which contains only the word "Karen". If you immediately choose Clips/Cut, the wave will be reduced to "I love" and the clip will be discarded. If you instead choose Clips/Copy and then choose Clips/Cut, the wave will still be reduced to "I love", but the clip containing "Karen" still exists. This clip could then be pasted back into this wave, say between the "I" and the "love", producing "I Karen love". Since the clip was copied, it could be pasted again - even though the wave has changed. If the new paste point was at the very end of the wave, the result would be "I Karen love Karen". To bring this clip into another wave, activate the other editor and select Clips/Edit. Now, change the active clip for this editor by clicking on the "Karen" clip, and close the dialog. Finally, select Clips/Paste and insert the "Karen" clip where needed. Helpful hint: it is possible to paste a clip into an empty editor window. This is how you can turn clips into waves.

To take this to its extreme, the "Karen" clip can be pasted into an empty editor. This creates a new wave which just contains the word "Karen". You could then create a clip of the "K" sound and repeatedly paste it to the start of the wave, thus producing "K-K-K-Karen". Assuming that a third editor contained the spoken words "Dreaming is great", you could create two more clips, "is great" and the "gr" sound. If you paste "is great" at the end of the wave, you get "K-K-K-Karen is great". Finally, repeated insertion of "gr" would yield "K-K-K-Karen is gr-gr-great" (which would undoubtedly make Karen happy to hear it). If you're thinking that you could make a clip of a cycle or two of a waveform, and then repeatedly paste it in order to create a brand new wave, you are exactly right. You would, however, be wasting a good deal of time since we have a Replicate function to do exactly this sort of thing with ease.

Backups

The **Setup/Backups** menu item lets you set how many levels of backup (i.e., undo depth) you want. It opens a small dialog with three choices: None, One, and Multiple. With Multiple, you get to type in just how deep you want it to be. Checking **None** removes the Undo capability. There is no backup. The advantage here is that there is minimal memory usage and processing overhead. Checking **One** gives a single backup. It's also very straight-forward in that if you hit the Undo button a second time, you undo the undo (ie, a redo). With **Multiple**, you can have several edits in reserve. Although this is very convenient, it also requires greater memory usage, which may slow processing. In general, it's probably best to keep this value below 10 unless you're working on small files and have a computer with a lot of memory. Note that you must use the Redo button to undo an undo when using Multiple. Redo is right next to the Undo button, but rotating clockwise instead of counter-clockwise. If you're wondering what the Redo button does if you only have One backup, it acts as a redundant Undo.

Undo History

The **Edit/Undo History** menu item allows you to jump immediately to a prior edit. It is used in conjunction with the Backups item.

The Undo History dialog shows a list of recent edit operations with the newest one at the top. Your current position in the undo/redo list is highlighted. Simply double click on the operation you'd like to go forward or backward through. It will "unwind" or "rewind" all the operations in between your current position and the selected position. If you Undo part way down the list and then perform a function (like EQ for example), the backups above the current point are no longer valid and are discarded for you automatically.

Here's a quick example: Say you load a sound, do a Gain scale, do Bass&Treble EQ, and then some AM. If your backup depth is at least three, the Undo History will list the following:

AM
EQ: Bass & Treble
Gain

"AM" will be highlighted. If you hit "OK", you will remove the AM effect. If you immediately reopen Undo History, you'll see the same list, only now "EQ: Bass&Treble" will be highlighted. Double clicking AM will Redo back to the AM effect, while double clicking Gain will move you back to original wave you loaded (undoing both EQ and Gain). It is possible to hit Undo/Redo during playback for comparisons, and yes, you can even call up Undo History during playback. This is particularly useful when using Play Affect since it loops (hit Spacebar for shortcut). If you're using the Show Position bar during playback, an extra bar will be drawn where the Undo/Redo kicked in, after the wave is redrawn. Although useful, this can get a little "busy" looking after several Undo/Redo's. A slight resizing of the window will get back a nice clean waveform drawing if you find this distracting.

Signal Processing Functions

Sample Wrench's digital signal processing functions allow you to manipulate waves directly in the digital domain. Some of the functions are modeled after studio processing effects devices that musicians have come to know and trust, such as a compressor. Generally, all DSP functions work in a similar manner. After selecting the desired item from the **Functions** menu, you will be greeted by a function dialog. Parameters may be set via pushbuttons, sliders, or text boxes. Unless otherwise noted, all numeric string values should be whole numbers (no fractions) and be free of imbedded spaces, commas, or the like.

After you've set the desired parameters and clicked on the OK button, the mouse pointer will turn into an hourglass and the progress bar will advance. After a certain amount of calculation, the new wave will be redrawn for you and the pointer will return to its default shape. The elapsed processing time will appear along the main status bar. Different functions have different calculation speeds, so don't expect the same timings for different functions. Also, calculation time is directly proportional to the size of the wave. The longer the wave is, the longer it will take to calculate the function. Like their real-world hardware counterparts, these "software circuits" will clip and distort a wave if they are overdriven. You can opt to ignore the process and continue with Wrench by clicking on the associated Cancel button. In general, dialogs open with the last settings you used so that it is very easy to reapply a function or effect. Finally, many functions also have presets.

Using Presets

Presets come in two flavors: **internal** and **external**. You can think of internal presets as "factory presets" and external presets as "user presets". The internal presets are always available and are accessed via the **Presets drop-down box** (which is normally found at the button of the effect's dialog). You simply select a preset from the list and all of the effect's parameters are updated accordingly. If you come up with a set of parameters you'd like to keep for later re-use, you can use **Save Preset** to copy it to your disk drive. This is an external preset, and you can create as many of them as you'd like. To recall one of these presets, use **Load Preset**. You can also give your presets a name (like the internal presets have) by typing it into the **Presets box** before saving. If you create many presets there are a few things you can do to better organize them. First, Wrench allows you to set a **Presets Directory Path** (under the Setup menu). Wrench will search this directory for appropriate presets and add them to the drop-down list for you. This is very convenient if you keep all of your presets in one place. This directory is also where the Load Preset file dialog will start. Second, the Load Preset file dialog defaults to searching for files with a **.dwp* extension** (dwp = dissidents wrench preset). If you always start the extension with these letters, finding them will be easy. We suggest that you add extra letters which convey the type of preset (for example, .dwpeqp for EQ:Parametric presets, .dwpflange for Flange presets, .dwpresyn for Resynthesize presets, and so on).

Click and Pop Removal

This is very simple to use and quite effective. It removes nasty transients, notably the clicks and pops of vinyl albums. To access Click and Pop Removal, select **Functions/Click and Pop Removal**. It has three settings: Aggressive, Normal, and Conservative. Aggressive will identify the largest pops but has the greatest potential for producing unwanted artifacts. **Conservative** will not remove the larger pops but does create the smoothest result. **Normal** is a nice middle ground, useful for the average case. Extreme pops will not be removed by this function but will probably be attenuated somewhat. These may be removed by drawing them out with the pencil tool (i.e., freehand draw) once this function has gotten the majority of the other pops and clicks.

A step by step example using Click and Pop Removal may be found in the Cleaning Samples section of the Tutorial.

Clone Wave

As its name suggests, Clone Wave creates an identical copy of a given wave. A new editor window will be opened for you, with this copy inside of it.

Combine Samples

This function serves as a mixer. It lets you join any two waves together. You can set the relative levels of the present wave and the import wave, as well as specifying an offset position. You can also perform a simple append operation, where one wave is joined up to another in sequence.

To call up the dialog, select **Combine Samples** from the **Functions** menu. First off, you must set the wave you wish to import using the **Import (Combine Present With)** list. Your choices are any of the loaded editors. It is quite possible to combine a sample with itself, if desired. You must now decide whether the imported wave will start ahead of or behind the present wave in the final result. The **Lags/Leads** buttons set this. This is only important if you set a non-zero Position offset value. This offset is specified by the **Position** text box. The value can be a straight offset, as set in the text box, or the text box can be used to call up a previously set marker. For marker usage, click on the **Marker Number** button under the **Position Using** group. Remember, you can use any one of the markers associated with this wave, but the one you choose must be set with Set Markers before it can be used here. The final controller is the **Import/Present mixing slider**. This adjusts the relative balance between the two waves.

A couple of helpful tips: If you would like to create a well controlled fade in/fade out from one wave to another, tailor the start and end portions of the wave with the Envelope Generator first, and then combine the sounds with an appropriate offset. By using the EG function, you can create very complex cross fades unobtainable on most systems. By doing this, you could get a guitar to fade into a flute, and then back to a guitar, for example. (Get used to combining different functions, the results can be very interesting, and the process very powerful). For an append, simply set the Position value greater than the size of the wave.

Compressor

If you have used signal compressors or limiters before, you know exactly how to use this function. Its purpose is to produce a leveling of the wave. In other words, it constricts the dynamic range of a sound. It can also be used to accentuate the attack portions of a sound (eg, a bass guitar). Within Wrench, it can be used to help smooth out sounds as a looping aid. For very specific and tight control over a wave's loudness contour, the Envelope Generator function is recommended in its place.

To call up the Compressor dialog, select **Functions/Level Control/Compressor**. This dialog contains five sliders. These sliders control the standard parameters **Threshold Level**, **Compression Ratio**, **Attack Time**, and **Release Time**, along with **Detection**. **Threshold** indicates at what signal level compression starts. It is adjustable from 0 dB (maximum signal) down to -60 dB. Unlike a hardware compressor, 0 dB is used to represent the absolute maximum output level. Also, this is an effective (RMS, which equates to loudness) value, so it will not perfectly match the level displayed by the vertical -dB scale. For example, a wave with a peak value of -10 dB may have an RMS value of -16 dB, it all depends on how dynamic the waveform is (the RMS value will always be more negative than the absolute peak). **Ratio** sets the degree or severity of compression. It is adjustable from 10:1 down to .2:1. A ratio of 10:1 means that a signal increase of 10 dB will be turned into a signal increase of 1 dB. Unlike many hardware compressors, this function offers fractional ratios which indicate expansion. A ratio of .2:1 means that a .2 dB input change will produce a 1 dB output change. Be careful with expansion. It is very easy to produce clipping if you don't watch out. **Attack** sets how long it takes for the compression action to kick in once the input signal exceeds the threshold. For example, you may want very quick transient peaks to pass through uncompressed. In a similar manner, **Release** sets how long it takes for the compression to cease once the input signal has dropped below the threshold. The Attack is adjustable from .01 milliseconds (10 microseconds) to 20 milliseconds. Release is adjustable from .01 seconds (10 milliseconds) to 2 seconds.

Precise values depend on the material being compressed and the effect desired. Inappropriate settings may produce an effect called "pumping" or "breathing", where the change in signal and background noise produces a noticeable effect. For general purpose work, ratios in the 2 to 5 range work well, as do thresholds in the -30 to -15 dB range. For attack and release times, consider values in the 1 millisecond and 100+ millisecond areas, respectively. (This can vary quite a bit, so we strongly recommend that you pick up a book on recording techniques if you are unfamiliar with this). Also, for you more experienced users, this function produces a gain change gradually rather than abruptly, like some hardware circuits. If you use the abrupt type, you may wish to compensate for the difference by reducing the threshold setting a tad.

Finally, the **Detection** parameter is unique to this compressor. This sets the dynamic sensitivity of the compressor, and may be varied between 1 and 50. Small values produce a more peak responding response, while higher values produce a longer term RMS response. Good general purpose values run between 10 and 20. Very small detection values are useful if you need to tightly track rapidly changing high frequency (high pitch) sounds. Large detection values are useful for slowly changing and low frequency (low pitch) sounds. Do not use small detection values on low pitch sources (such as bass guitar) as this may create some distortion of the sound.

The Compressor also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Crossfade Looping

The one headache many people have with samplers is achieving seamless loops. There are many techniques which can be used. Some of the ones supported here include zero crossing detection, compression, and custom envelope generation. As useful as these techniques are, they will not take care of all waves. Crossfade Looping is designed to do nothing but smooth out loops. It does this by carefully mixing sound from one part of the loop with sound from the other part.

To use this function, select **Functions/Looping and Keymaps/Crossfade Loop**. The **Fade Area group** lets you set one of three crossfade formats. **Start** takes sound from the end of the loop and mixes it in at the loop start. **End** takes sound from the front of the loop and mixes it in at the loop end. **Dual** mixes sound from the start and end at both points. If you don't care about the release portion of the wave, End is a good choice for a sustain loop. Dual is the most drastic, but offers the greatest consistency. The sounds can be mixed in either a **logarithmic** or **linear** fashion with the **Fade Type group**. You can specify if you want to crossfade either the **sustain** loop or the **release** loop with the **Loop group**. The **Fade Amount** text box sets how much sound on either side of the loop start and end will be mixed together. This value can never exceed one half the number of samples between the loop start and end points (don't worry, if the value is too big, Wrench automatically figures out the maximum size. If the loop is very large, be careful not to use too large of a Fade Amount. Doing so could adversely effect the attack transient with the Start and Dual crossfade types. Since the tail end of a wave can be shorter than the crossfade amount, you can force the crossfade to speed up so that it doesn't abruptly stop (the same is true for the lead in portion of the wave). To enforce this optimization, select the **Tailor Edges** check box.

Please note that some sounds will never be successfully looped. Any sound which changes a great deal from start to finish, or which contains a very complex harmonic structure, may simply refuse to co-operate. The whole process of looping assumes that there is a certain amount of consistency in the wave. Without it, your ears will always pick up on the timbre changes. It has been said that looping is half science and half art. The science is in knowing how to best use the looping tools. The art is in picking likely loop points to begin with.

Cut / Keep List

This is a great function if you need to chop a wave into segments, or remove several chunks from a wave. Although these processes can be performed using standard cut and paste techniques, Cut/Keep List is much quicker. To use this, select **Functions/Cut Keep List**.

Using markers, you can specify chunks of a wave to

1) **Save** to the Multi-Clipboard,
2) **Save** to disk,
3) **Cut** from the wave (creating a new wave in a new editor with the original intact).

Basically, you specify an area to be cut or kept by bounding it with a pair of markers. This is very handy in conjunction with Wrench's "drop markers on the fly" capability. Simply Shift+click to set the markers, one pair per area. The chunks are processed by marker position, not marker ID, so it matches what you see on the editor window. If you select **Save to Disk**, then each area will be saved to disk using the base name plus a sequence number. If you select **Save to Multi-Clipboard**, then each area will copied and placed into the Multi-Clipboard, again using a sequentially numbered name. If you select **Cut**, then the markers are processed as a cut list, meaning that the areas will be cut from the wave. In this case, a new editor will open with just the remaining portions.

In all three cases the original wave is left intact. The **Preview** button will play the areas to be saved or copied for cases one and two, and the remaining wave for case three.

DC Offset

This function adds a constant shift to the wave. Generally, you don't wish to have any DC (direct current) in a wave. A pure DC wave would be shown in an editor window as a simple horizontal line shifted either above or below the 0 axis. The purpose of this function then, is to add an opposite offset to cancel any existing offset. Waves may have DC offsets as a result of the sampling process if the A/D converter and filters used aren't up to snuff. Certain editing functions (most notably asymmetrical transfer functions and rectification) may also create offsets. Selecting **Functions/DC Offset** brings up the small DC Offset dialog. There is one numeric box present which allows you to set the offset. The offset is in percent from -100 to +100 for Wrench 24/96 (32 bit data), and in quantization value from -32768 to +32767 for 16 bit data Wrench. Values are generally much less than these ranges. Small DC offsets are difficult to spot since the natural asymmetry of a wave tends to hide the shifts. If you type in 0 for the value, Wrench will calculate the offset needed. This will only work if the offset is rather small; excessive offsets or very bizarre contrived waves may produce an erroneous value.

Delete (Remove)

As its name suggests, this will remove the edit area from the sample (ie, cut it out). This is basically the same as Cut, but without copying the chunk to the system clipboard.

Envelope Generator

Envelope refers to the over all loudness contour of a sound. Early synthesizers allowed the envelope to be set through the use of ADSR circuits which varied the attack, decay, and release times, as well as the sustain level. While very useful, Wrench has come a long way from that humble beginning. Wrench lets you draw a desired envelope gain characteristic with the mouse, which it then applies to the wave. You can implement both subtle and overwhelming envelope changes. One of the unique characteristics of this function is that you get to draw a gain curve based on decibels. Thus, constant fade in or fade out rates are drawn as straight lines rather than compound curves. Many helpful tools are also available.

Before continuing, it is important to remember that you draw a super-imposed gain characteristic for the wave, not the desired resulting envelope (these two will look the same only if the original wave has a constant loudness). To make an analogy to those older synths, you are effectively drawing the final VCA's control voltage waveform.

The envelope drawing area takes up the majority of the EG window. Drawing the envelope characteristic is much like using the Free Hand Draw mode. To activate drawing, press and hold the left mouse button. Moving left to right draws the desired EG shape, while right to left motion erases it. The gain characteristic is calibrated in dB. Each vertical line represents .25, .5, or 1 dB of gain, which results in ranges of +/- 15, +/- 30, or +/- 60 dB. The middle line always represents 0 dB (a gain of unity). Positive dB values are gains, while negative values are losses. +20 dB represents an increase by a factor of 10, while -20 dB represents a reduction by a factor of 10. (If you are not familiar with dB notation, it is a logarithmic scale. 40 dB represents a factor of 100, 10 dB a factor of a little over 3, and 6 dB a factor of 2). The calibration is controlled by the **Scale: dB/Step group**. Near this is the **Zero It** pushbutton. Selecting this will null the EG characteristic, returning all values to 0 dB. Like most Wrench functions and effects, the EG dialog remembers your most recent EG shape. It will be there the next time you call up this function.
For general purpose work, you will probably find the .25 and .5 dB scales to be the most useful. The 1 dB scale can sometimes produce "stair step" type effects on waves, particularly if you draw very gradual slopes. The EG function can be put to good use with the standard applications of altering attack and decay times, as well as sustain levels (eg, pronouncing the attack and decay section to give a bass guitar more bite or pop). It can also be of great use when looping sounds. It is not uncommon to create a loop which is "clickless", but which suffers from a "thump". Thumps are usually caused by a variation in loudness. For example, you might be trying to loop a guitar wave. Once plucked, a guitar string will vibrate with ever decreasing magnitude, getting progressively quieter. Even if the chosen loop points are at zero crossings and have similar surrounding shapes, the volumes at these points will be different. The sonic result is a loop thump. If you apply a slightly rising EG shape, you can compensate for the natural fade out character of the guitar. By keeping the overall volume constant, the thump will be drastically reduced. (As a side note, some people record the original wave with a compressor to achieve a similar consistency. While Wrench does offer a compressor function, with a little practice, you will probably discover that the EG function is

quicker for this purpose). Finally, even if a wave does not fade out (eg, a woodwind), it may have certain irregularities which produce thumps. These bumps and dips may be smoothed over with the EG function by applying a reverse characteristic (ie, draw a dip in the EG to compensate for a wave bump, and vice versa).

The final set of buttons allow you to manipulate the gain curve. They are: **Invert**, **Shift Up**, **Shift Down**, **Reverse**, and **Trace**. Selecting **Invert** will flip the present curve about the 0 dB axis. In other words, what used to be dips in the envelope become peaks, and vice versa. **Shift Up** moves the entire curve up, this increasing the overall gain. **Shift Down** moves the entire curve down, thus decreasing gain. **Reverse** flops the curve from back to front. Selecting **Trace** will create a curve which approximates the envelope of the wave segment you are presently editing. This function uses true peak to peak detection when it calculates the envelope in order to obtain the highest accuracy. Remember, the gain curve is drawn in a decibel (logarithmic) format, so its shape will not be identical with what you see in the editor window (linear format). The decibel format is convenient because it gives you greater control over low level signals. The extracted envelope is always calculated using the .5 dB per step scale. In this way, you can quickly apply it to a waveform with extra emphasis by using the 1 dB scale, or in a more subtle fashion with the .25 dB scale. Here are a few good examples of using Trace and Invert.

Suppose that you would like to impart the natural decay of a piano note to a static organ sample. First, load the piano sample into an editor and then select the Envelope Generator function. From the EG window select Trace. In a moment, the envelope of the piano will appear. Save this curve using Save Preset and then exit the EG function by hitting Cancel. Now, load the organ sound into an editor and again select Envelope Generator. When the EG window opens this time, use Load Preset to load the curve you saved in the last step. You can apply this shape to the organ sample by simply hitting the OK button. That's all there is to it.

If you wish to accentuate volume variations, simply apply a sound's envelope to itself. For example, you might want to increase the popping sound of a bass guitar. If you Trace the sound sample, you will note that it starts off high and then begins to decrease. If you apply this curve to the bass guitar itself, you will end up making it decrease at a faster rate, and thus, increase the relative intensity of the beginning pop. In a similar manner, doing this to a sample which has natural tremolo will increase the depth of the tremolo.

Another interesting application is to use both Trace and Invert on a single wave in order to smooth out volume variations. This can be useful for certain samples which are difficult to loop. By inverting the envelope, you will be giving extra gain to low level areas, and less gain to high level areas. The end result is an overall constant volume. The process is fairly simple. Once the EG window is open, select Trace. Once the new curve is drawn, select Invert. Now select OK to apply this mirror image curve to your wave. The end result will be a sample with a fairly stable volume contour. This can be very effective to bring out quiet portions of recorded speech.

The Envelope Generator also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Equalization

Signal equalization (EQ) allows you to alter the frequency spectrum or timbre of a sound. The simplest type of equalizer component is the filter. A filter simply removes certain sections of a sound. Generally, this means that it will remove high or low pitches, as in a scratch or rumble filter. A step up from this is the shelving type of equalizer. Good examples of shelving equalizers are the bass and treble controls found on most stereos. These allow you to boost as well as cut certain pitches. Graphic equalizers are similar but offer several frequency bands worth of control. One of the more advanced and flexible forms of EQ is the parametric. Parametrics allow not only boost and cut, but control over the exact tones affected as well. Sample Wrench allows you to use all four types of EQ. All four types include the standard selection for internal Presets, and the ability to Save or Load external presets.

EQ: Bass/Treble

This dialog is split into two sections, a bass control and a treble control. They can be used independently or together by selecting the **Use Bass** and **Use Treble** check boxes. Each section contains a **cut/boost slider** with a 20 dB range and a **Frequency** text box. Unlike the bass and treble controls found on most stereos, Wrench's controls are more flexible. You can set precisely where you'd like the cut or boost to begin. For example, in order to increase the thump of a bass drum without making vocals sound too chesty, try setting the bass frequency to around 70 Hertz. General purpose values would be a few hundred Hertz for bass and several kiloHertz for treble.

EQ: Filters

This dialog is split into two sections, a High Pass filter and a Low Pass filter. They can be used independently or together by selecting the **Use High Pass** and **Use Low Pass** check boxes. Each section contains a list of possible **filter orders** (attenuation rates) and a **Frequency** text box.

High pass filters allow high frequencies through, and limit low frequencies. A rumble filter is an example of a high pass filter. Low pass filters by contrast, limit high frequencies. The order of a filter determines how severe the filter action is. Higher order filters produce a more pronounced filtering action. Your choices from the Order group are 1st, 2nd, and 5th. For the technically minded, the 1st and 2nd order filters use a standard Butterworth alignment and roll off at 6 and 12 dB per octave, respectively. The 5th order system uses a .5 dB ripple Chebyshev alignment producing over 40 dB per octave initial roll off. The 1st and 2nd order systems are recommended for general purpose work, while the 5th order is primarily intended for re-sampling or special purposes (see the section on Sample Rate Transposition). Due to its complexity, the 5th order filter takes somewhat longer to calculate than the 2nd order filter.

EQ: GraFreq

This dialog contains five individual graphic sections. Each section contains a **cut/boost slider** with a 20 dB range, and a **Frequency** text box. Unlike typical graphic EQ where you're forced to choose from a selection of fixed frequency bands, GraFreq is really a semi-parametric EQ since you can specify the exact frequency you like to cut or boost. The range of control for each band is about 1.5 octaves.

Note that the frequencies do *not* have to be in ascending order, nor must they be equally spaced. Also, you can elect to use fewer than all five bands. To do so, simply set the associated cut/boost slider to 0.

EQ: Parametric

This dialog contains two individual parametric sections. They can be used independently or together by selecting the **Use A** and **Use B** check boxes. Each section contains a **cut/boost slider** with a 20 dB range, an **Octaves slider** which varies from .1 to 3, and a **Frequency** text box.

Parametrics are sometimes referred to as universal EQs since you have control over all parameters. Unlike graphic EQ where you're forced to choose from a selection fixed frequency bands, a parametric allows you to specify the exact frequency you like to cut or boost. You also have control of how much material on either side of this frequency is affected by using the Octaves slider. A setting of .1 is very narrow and is useful for making things like notch filters. At the other extreme, a setting of 3 octaves is very broad and affects a wide range of nearby tones.

EQ: General Comments and Examples

The minimum setting for Frequency in all EQ sections is 5 Hertz and the maximum setting is 95% of the nyquist rate. (The nyquist rate is one half of the sampling frequency, Fs). Here are a few examples:

Let's say that you've recorded a flute, and there is a subtle but annoying low frequency rumble in the background. How might you get rid of it? One way would be to use a high pass filter. You might set the Frequency for 100 Hz, and the Order to 2nd. Example number two, assume that you've recorded a tom-tom, but you'd like it to have a bit more edge and attack. To do this, consider using the parametric section. Frequency might be set to the 2000 to 4000 Hertz range, with the Octaves set to perhaps 1.0. The Boost/Cut slider might be set for 3 to 6 dB of boost, if you only need a moderate increase in attack. Only a single section would be needed. Example number three, you need to increase the sizzle on a hi-hat. A treble shelving EQ would be a good candidate here. Frequency might be set in the 5000 to 10000 Hertz range, depending on the character desired. If only a subtle change is needed, the Boost/Cut slider might be set for 1 to 3 dB. Mind you, all of the above values are just ballpark numbers. Actual values vary considerably depending on the instrument used, how it was recorded, and the exact effect desired.

Fast Fourier Transform Spectrum Analysis

The FFT is used as an analysis tool. It does not change the contents of the wave. In essence, this function lets you examine the frequency spectrum of the wave as a function of time. You have a choice of several different graphs styles and can save the graph in popular graphics formats. You can also save a text file of data values. The window graph is either a pseudo-3D graph (waterfall style) or a 2D graph (sonogram style). The 3D style plots frequency, signal strength, and time using a projection which you can view from any angle. The 2D graph plots frequency versus time and shows amplitude as contours and color coded zones. The data file option produces a table of frequency and signal amplitude pairs. You might use the FFT in conjunction with the Equalizer function in order to determine where to place a high frequency "hiss" filter, for example.

To use the FFT, select **Functions/FFT**. You may specify the **resolution** of the analysis in six gradations, from 2048 points to 64 points, by clicking on the associated radio button. A 2048 point FFT has very high frequency resolution with respect to the 64 point FFT. In other words, it can discern a greater number of individual tones. It also takes a lot longer to calculate. The 64 point FFT, however, provides greater time resolution. You can track spectrum changes more closely with this type. Due to the nature of the FFT, you cannot achieve high resolution in the frequency and time domains simultaneously. The current edit Affect area is used for the analysis, up to a maximum of 1000 "time slices".

The graph has its own window. The title bar will indicate which editor the FFT is being calculated from (note, you cannot have more than one FFT window open at any given time). This window can be moved, depth arranged, and resized. Also, it can moved independently of the main Wrench application. Wrench will calculate and display the graph one slice at a time. The **File** menu offers various save options including saving the graph itself in either JPEG or BMP formats, saving it to the Windows clipboard, and saving the graph data as a text file. You can also send the graph to your printer. The **View menu** includes the four main **Draw styles** (for convenience) and the **Options** choice. Selecting Options brings up a dialog which allows you to change numerous aspects of the graph. More on this in a moment. The final menu brings up Help on the FFT.

Draw styles and options

Variations on the graph are accessed through **View/Options**. The Options dialog is opened automatically when FFT is first called up. A "right click" context menu will also lead to the Options dialog. There are really two major types of graphs: 3D and 2D. You get 3D if either **Draw Mesh** or **Draw Shaded** are selected. Mesh alone sort of looks like a net, with Shaded adding a surface effect. **Draw Contours** adds elevations (like a topographical map), and **Draw Zones** adds color to the elevations. If only Zones and/or Contours are/is selected, you get a 2D style graph which looks just like a topo map (commonly used for speech research). With four base styles, there are a total of 15 different combinations (deselecting all four gets you a lovely blank graph).

There are many variations including background color, font size and style, whether to draw the various axes, and so forth. **Project Zones** and **Project Contours** only work for 3D styles. Project Zones will place a 2D zone graph under the 3D graph while Project Contours will place a 2D contour graph above the 3D graph (i.e., in the time/frequency plane). **High Resolution** offers highest accuracy but can get busy to look at (and wait for), especially for Linear Frequency axis. It's great for Log axis and zooming, though.

For the terminally curious, here's the fun part: you can zoom/pan/rotate and generally go nuts with the graph. Be forewarned that the interactive nature of the graph can be adicting. Most of the following operations require the "middle" mouse button. If you don't have a middle button, use left+right together- be careful though, because sometimes a 2-button mouse may require that one button (typically right) be engaged/disengaged before the other in order to properly mimic the middle button. An outline of the graph will be shown during movements.

To ZOOM: Hold CTRL and draw a zoom box with the left mouse button, just like in normal Wrench windows. You can also hold CTRL and press the middle button to zoom in/out by moving the mouse. (Release button before releasing key.)
To PAN: Hold SHIFT and press middle mouse button, moving mouse in desired direction.
To ROTATE: Press middle mouse button and move mouse (see below for axis rotation).
To RESET VIEW: hit the R key.

You can **independently rotate about each axis** by depressing the middle mouse button, hitting the appropriate letter, and moving the mouse. Use X or T for Time axis, Y or F for Frequency axis, Z or A for amplitude axis. You can also use E for the "eye" which is sort of like using a joystick. Note that you can do this in sequence, for example, first hitting F, rotating, hitting A, rotating, and so forth. When you finally release the mouse button, the graph is redrawn. Along with the graph outline, the chosen axis of rotation will also be drawn.

Please note that large analysis areas can take a while to plot, especially in high resolution or 3D style. Generally, it's best to stay in low resolution until you get what you like, then switch to high resolution for final examination. There is a 1000 record limit on the analysis (in case you accidentally try to do an FFT series on a 10 minute song!). Also, for stereo signals, the left channel is analysed. If you want the right channel, make sure that Edit Right is turned on and Edit Left is turned off.

A few notes and thoughts for the technically minded: Wrench uses a real-value scheme with 50% record overlap. The data is adjusted first with a Hamming window. This function is intended for general purpose music work only. For users with more demanding graphing and analysis requirements, we suggest the data file option. This information can then be examined manually or loaded into some other analysis or graphing program. The format of the data file is straightforward. At the start of each record will be a line stating the current record number. After this will come two columns of floating point values. The first column is the frequency in Hertz and the second column is the amplitude. The range of values for the amplitude is from -32768 up to +32767 (i.e., a range of 2 to the 16th power) for 16 bit integer data, and +/-1.0 for floating point data.

Gain

Sometimes you need to scale the amplitude of wave (or a portion of a wave). The Gain function allows you to do this. Selecting **Functions/Level Control/Gain** brings up the Gain dialog which contains a slider. This slider is used to adjust the gain from -30 dB to +30 dB, in .1 dB steps. As a reference, a 6 dB change represents a scaling by a factor of two, while a 20 dB change scales by a factor of ten.

Interactive Loop Window

The Loop Window function takes the drudgery out of setting loop points. Selecting this function opens a special window which looks rather like a normal editor window. Like an editor window, a portion of the wave is drawn in the window, and zoom in/out buttons are located in the window toolbar. The difference is that the display is actually made up of two adjacent displays, the one on the left showing the loop end area, and the one on the right showing the loop start area. By allowing you to see both ends of the loop simultaneously, you can match the loop points more precisely to create a seamless loop. You have complete control over the exact zoom level of your view, and the vertical orientation as well. You will note that there are a set of left/right buttons and a horizontal slider for each half of the display. These items control the positioning of the loop start and end points. Initially, a moderate zoom level will be used with large waves so that you don't have to start zooming in yourself (about 1000 sample points will be visible).

The toolbar contains buttons indicating the loop chosen for editing (these look like standard Wrench loop brackets, S for sustain, R for release). If no loop is set, nothing will be drawn. (ie, if you select Sustain but no sustain loop has been created, the display will be blank). There are also buttons for setting the display style. Two forms are available: **Simultaneous** and **Spliced**. When **Simultaneous** is chosen, the loop point for each half of the display will be drawn in the center of each display. It will be denoted with a vertical line. This style lets you see the wave data immediately before and after both loop points. In contrast, **Spliced** only draws the data prior to the loop end, and after the loop start. This places the vertical end line at the right end of its display, and the vertical start line at the left end of its display. The result is that the vertical start and end lines coincide smack in the middle of the window. As a result, you can see exactly how the end of a loop "flows" back into the beginning of a loop. It's also a very handy way to make sure that the start and end levels are identical. If this description sounds a little confusing, simply examine the display as you switch back and forth between Simultaneous and Spliced. **Play** and **Stop Play** buttons are also included in the toolbar for your convenience.

In order to change a loop point, simply click on the associated left or right arrow button. For large changes, you can move the appropriate horizontal slider. Note: If you make a drastic change, such that the loop end is pushed before the loop start, the positions will be automatically tracked and readjusted for you. In order to help you keep track of the loop points, their numeric positions are printed out directly above the display.

The final item of interest here is the **Auto Redraw** button. This is important: The Loop Window interacts with its parent editor window. In other words, if the editor window has Show Loops enabled, any change to a loop in one window will be reflected in the other window. If you change a loop point in the Loop Window, the parent editor window will be updated automatically to show this new change. Sometimes, this can slow things down a bit, so you have the option of turning this feature off by deselecting Auto Redraw. By the way, you loop freaks out there will be happy to learn that changes in the parent editor window are reflected in the Loop Window. This means that you can grab the loop points in the editor window using the mouse, and the Loop Window will be updated accordingly. Consequently, you can position the loop points by using both windows simultaneously, each display tracking the other automatically.

Key Map (Simple)

The Keymap item lets you set the **root key**, **high key** and **low key** for the wave in terms of MIDI sample numbers. The Keymap dialog also allows you to set a fine tune amount for waveforms. This is useful if you recorded an instrument which was slightly out of tune. The range for **Fine Tune** is between plus and minus 50 cents (where 100 cents equals one halfstep). This info may be transferred to a sampler, depending on the capabilities of the sampler. For example, fine tunings are not supported under the sample dump standard, but are supported under the SMDI protocol. Also, the Fine Tune amount is used in the audio playback modes. If you prefer, Fine Tune can be adjusted "by ear" using the MIDI Keyboard Window. You can also use the MIDI Keyboard WIndow to graphically enter keymaps, if you prefer.

Maximize (Scale To Full)

Now here's a straight forward function. Maximize will give your sound as much gain as it can without producing clipping. You end up with maximum volume. There is no need for a dialog here, selecting **Functions/Level Control/Maximize** does its job, and that's it! As in hardware, good, strong signals are a must if you want to keep noise levels low. Be fore warned though, just like hardware, no amount of gain will reduce noise once it has found its way into the signal. There is a common misconception that editing in the digital domain is noise-free. It's just not so.

Finally, since floating point data (Wrench 24/96) can be overscaled without clipping, Maximize can be used to bring an overscaled wave back to normal range (i.e., 100%). This is very important because overscaled waves will be rendered as clipped waves during preview or when saved out in formats other than WAV32.

MIDI Keyboard (with map)

The MIDI Keyboard window is used for three things: you can hear different pitches using the mouse, you can trigger a remote MIDI device, or you can set your keymap graphically. You open the keyboard by selecting **MIDI Keyboard..** in the **Functions** menu. There are two areas of interest in the window. The upper area is comprised of two claviers which together span the entire MIDI note number range of 0 through 127. The lower area is comprised of a set of buttons and other gadgets. The **Fine Tune slider** will show the present tuning, and the values for **Low**, **Root**, and **High note** will be printed next to their respective buttons. Also, these values will be reflected on the two claviers. Small colored squares will be placed on the keys to indicate the low, root and high values. The low note square will be placed at the bottom of the key, the root note square in the middle, and the high note square at the top. Normally, these squares will all be different colors making identification even easier (unless you're running with very few colors or in monochrome). You can change the keymap by using the mouse. To do so, select which of the three values you'd like to alter by clicking on the appropriate button. At this point the mouse pointer will turn into the word "To" as you pass it over the claviers. Now, click on the key you'd like the note set to. Once this is done, the text and colored squares will be updated to reflect your change. As a side note, if you'd like to set two or three of the values to the same key, simply click on the High/Low/Root buttons in sequence before clicking on the desired key.

In order to listen to a wave, simply click down on the key you'd like to hear. Playback will start and continue for as long as you hold down the mouse button (assuming that you have a looped sample and not a one-shot type). When you release the mouse button, playback will halt. You can listen to different pitches by simply clicking on different keys. If you move the mouse over the keyboard during playback, the pitches will change also. Note that the new pitches will not restart playback from the beginning. In this manner, you can listen to the effect of transposition on loops very quickly. This is also useful for listening to pitch shift effects on very long samples. Remember, you can always restart playback from the beginning by simply releasing the mouse button and selecting a new key. If you move off of the claviers at any time, playback will halt.

You can also direct the output of the claviers to a MIDI device instead of to the internal audio circuits. To do this, simply set the desired **MIDI driver**, **volume** and **channel**, and then select **MIDI Triggers** from the **Clavier Output Type group**.

For playback, The Keyboard Window uses the **Fine Tune** value from the Keymap dialog, allowing you to hear small adjustments in pitch. You can adjust the Fine Tune amount directly and hear the result in realtime with the Fine Tune slider. To use this, simply click and hold on the slider's knob. This will start playback. As you move the slider, the Fine Tune readout will change and the pitch of the sound will start to shift. The pitch will increase as you move to the right, up to a maximum of 50 cents (one half of a half-step) from the nominal. As you move to the left, the pitch will drop by a maximum of 50 cents. When you release the mouse button, playback will cease, and the last value on the slider will be used for the sound. If you decide that you don't want any shift at all, make sure that you return the slider to 0 cents. A word of caution: due to the inherent limits of the computer's internal audio circuitry, small shift changes may not be audible and there may be some lag in time response. The exact limits depend on the pitch of the sound and its sampling frequency.

Please note that the keyboard will allow you to play samples which are way too high or too low for the computer's internal audio circuits to play properly. The result will be alias distortion and other sonic weirdness. This will not affect your waveform data, it just sounds strange and is not representative of the true sound which may be properly played back by a dedicated sample module or keyboard. Also, the mouse response at very high and low pitches may be somewhat sluggish, in that moving to a new key or releasing the mouse button will not change or stop the sound instantly.

Mono-Stereo

Mono-Stereo allows you to turn a mono waveform into a stereo waveform, or a stereo waveform into a mono one. For the second channel of mono waves, a source needs to be specified. You may specify any of the waves loaded in any of the editors (including itself). If the new waveform data is larger than the original, it will either be chopped or padded with silence to make it as long as the original. If you're starting with a stereo wave and you select this menu item, you will create a mono wave. The result can be either the left channel, the right channel, the sum of the two, or their difference. You can also choose to swap the left and right channels. Quick tip: Double-clicking on the desired list item is equivalent to selecting the item and then selecting OK.

Mute (Silence)

This function brings the edit Affect area to zero, thus producing silence. To avoid sudden turn-on or turn-off clicks, consider using the Smoothing feature (under the Setup menu) to automatically fade-out or fade-in to the silenced area.

Noise Gate

If you have used noise gates before, you know exactly how to use this function. Its purpose is to remove sections of a sound which fall below a certain amplitude. This can be used to tighen up instruments, remove background leakage, or even for special effects such as gated reverb.

To call up the Noise Gate dialog, select **Functions/Level Control/Noise Gate**. This dialog contains four sliders. These sliders control the standard parameters **Threshold Level**, **Attack Time**, and **Release Time**, along with **Detection**. **Threshold** indicates at what signal level gating starts. It is adjustable from -10 dB (below maximum signal) down to -90 dB. Note that this is an effective (RMS, which equates to loudness) value, so it will not perfectly match the level displayed by the vertical -dB scale. For example, a wave with a peak value of -10 dB may have an RMS value of -16 dB, it all depends on how dynamic the waveform is (the RMS value will always be more negative than the absolute peak). **Attack** sets how long it takes for the gate to open once the input signal rises above the threshold. For example, you may want very quick transient peaks to remain gated. In a similar manner, **Release** sets how long it takes for the gate to mute the audio once the input signal has dropped below the threshold. The Attack is adjustable from .01 milliseconds (10 microseconds) to 20 milliseconds. Release is adjustable from .01 seconds (10 milliseconds) to 2 seconds.

Precise values depend on the material being compressed and the effect desired. Inappropriate settings may produce no gating at all or may chop the "heads" and "tails" off of program material. For attack and release times, consider values in the 1 millisecond and 100+ millisecond areas, respectively. (This can vary quite a bit, so we strongly recommend that you pick up a book on recording techniques if you are unfamiliar with this).

Finally, the **Detection** parameter is unique to this noise gate. This sets the dynamic sensitivity of the gate, and may be varied between 1 and 50. Small values produce a more peak responding response, while higher values produce a longer term RMS response. Good general purpose values run between 10 and 20. Very small detection values are useful if you need to tightly track rapidly changing high frequency (high pitch) sounds. Large detection values are useful for slowly changing and low frequency (low pitch) sounds. Do not use small detection values on low pitch sources (such as bass guitar) as this may create some distortion of the sound.

The Noise Gate also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Normalize

Sometimes you need to scale the amplitude of wave (or a portion of a wave) to a precise level. The Normalize function allows you to do this. Selecting **Functions/Level Control/Normalize** brings up the Normalize dialog which contains a slider. This slider is used to adjust the peak signal level from 0 dB to -40 dB below clipping, in .1 dB steps. As a reference, a -6 dB change represents a scaling by a factor of one half or 50%, while a -20 dB change scales by a factor of one tenth or 10%. Note that the slider indicates the final value of the peak, not the amount of gain the signal receives. For example, no matter how quiet or loud a sound is, a setting of -10 dB will produce a sound with a peak value of 10 dB below clipping.

Noise Reduction

Noise Reduction is designed to reduce both broadband noises and noises with a unique signature. To access Noise Reduction, select **Functions/Noise Reduction**.

This has two parts, Noiseprint and Thresholding. **Thresholding** works by specifying a level below which signal components should be considered noise. This is very effective for cleaning up the quantization noise of 8 bit samples, or other broad spectrum noises. For unique noise signatures (like hum), you'll want to check out the **Noiseprint** based process. Here, you must identify a segment of just the noise you're trying to remove. Wrench will then attempt to suck this out of the signal, leaving you with a clean result. You can choose to use either or both processes simultaneously with the **Process** section. Simply check the modes you'd like.

The severity of both Noiseprint and Thresholding are controlled by the **Reduction Amount** buttons (Heavy, Normal, Light). **Heavy** is most Aggressive and will yield the lowest noise floor but artifacts may be produced such as filtering or phasing effects. This depends on the source material and the sample rate. The Heavy setting can produce up to a 60 dB noise reduction. The **Conservative** setting produces few artifacts but is limited to about a 20 dB improvement. The **Normal** setting is a good compromise yielding a best case 40 dB noise reduction. The **Tracking section** is used to compensate for changes in source material. For quickly changing material (fast dialog, percussion, etc.) use the **Rapid** setting. The **Steady** setting is suited for longer sustaining sounds.

Noiseprint Specific

For the noiseprint, you can use any of the open wave editors or you can load a file from disk. To use an existing editor, select **Wave Editor** from the **Noiseprint Source** section. The first quarter second or so of the sample will be used as the noiseprint. This means that you will generally do one of two things to obtain the noiseprint:

1) If the sample you're working on has a lead-in section containing just the noise, select the current wave editor as the source of the noiseprint. This is fast and convenient, but we aren't always so fortunate. If a good section of the noise can be found elsewhere (e.g., at the end of the sample), use method two.

2) Open an empty editor. After identifying a good noise chunk, select it with the mouse and paste it to the empty editor. Use this editor as the source of the noiseprint. If this noise occurs often (in several sound files), you may wish to save the contents of this "noise editor". You can then reload it as needed by selecting **External File** from the **Noiseprint Source** section of the dialog.

Thresholding Specific

The **Threshold slider** sets the specific level below which signal components are ignored. If the value is set too high, a side effect which sounds like excessive filtering can be heard. if the level is set too low, some noise will remain. The Threshold level is further controlled by the **Threshold Type** selection of Static or Dynamic. **Static** means that the Threshold is constant, relative to maximum output. **Dynamic** means that the Threshold is relative to the current signal loudness, so it moves up and down with volume changes. Generally, Static gives good all around results but Dynamic can be useful for Aggressive processing of certain kinds of material. Indeed, it is sometimes best to make two passes, once with each type.
A step by step example using Noise Reduction may be found in the Cleaning Samples section of the Tutorial.

Replicate

This function makes a given number of copies of the selected wave data and inserts them at the end of the original data chunk. You can specify the copies in terms of the actual number of repeats, or in terms of the total amount of time (in seconds or milliseconds) that you'd like the copies to occupy.

For example, let's say that you have a wave which contains 50,000 sample points. The edit Affect mode is chosen as Markers 0,1. Marker 0 is set to sample point 2000, and marker 1 is set to sample point 3400. If you select the Replicate function and set it to 12 repeats, the data from point 2000 through point 3400 will be replicated 12 times and inserted after point 3400. Another example of this would be to create a clip containing a single cycle of a wave, and then paste that clip into an empty editor. You could create a brand new wave based on this cycle by setting the function to create enough repeats to fill out, say, a three second wave. If desired, dynamics could then be added to the wave through the Envelope Generator and Equalizer functions.

Sample Rate Transpose

A common problem when transferring a sound from one sampler to another is a mismatch in **Fs**, the sampling frequency. If a wave is recorded at say, 30 kHz and then played back at 32 kHz, there will be a noticeable rise in pitch. As a matter of fact, this is how many samplers create new pitches, by varying the playback rate. When you are transferring sounds, this is undesirable as it means that your root pitch has changed. Even though the instrument was recorded at concert A, it may be playing back at C#. In preference to this, it is often desirable to recalculate a wave to yield a new effective sampling frequency. With a little insight, you can also create pitch transposition.

To call up the Sample Rate Transpose dialog, select **Sample Rate Transpose** from the **Functions** menu. Directly below the title is the present Fs value in Hertz, and next to this is the **New Rate** text box where you will enter the desired Fs. Many samplers give only approximate values for their sampling frequencies. It is not uncommon to find a unit which says it samples at 31 kHz, when in reality it samples at 31.25 kHz. Make sure you read the fine print on the unit's specs. It is also possible to enter the sampling period instead of the frequency. Period is just the reciprocal of frequency. Sometimes Fs is a repeating number (like 41.66666.... kHz), and the associated period isn't! If you want to use period, select Period from the Units group. The present period will be displayed instead of the present Fs, and the New Rate value will be taken as a period as well. The **Transpose Type group** specifies how the transposition is to be calculated. The default choice is **Linear**. This uses a simple linear interpolation scheme to create a new sequence of samples from the original. It is fairly quick, but suffers from noise and distortion if the wave has a fair amount of high frequency content. The second choice is **ReSample**. This uses a far more complicated and time consuming procedure. It does, however, produce the highest possible fidelity. The third choice is **Rate**. This doesn't perform any calculation at all, it just resets the Fs value. Some samplers will not accept a wave if the specified Fs is not available on that machine. Even if you don't care about the resulting pitch shifts, you may need to reset Fs to a matching value. (A fourth choice, Quick Pitch, is discussed below).

If you are down-converting, that is, going to a sample rate which is lower than the original, you should low pass filter the sample first in order to avoid alias distortion. You can do this automatically by selecting the **PreFilter** check box. Here is how to do it manually: The filter should be tuned to the new nyquist frequency (one half of the new Fs). For example, assume that the original wave's Fs is 42 kHz. It's nyquist rate is 21 kHz. In other words, the wave should contain no information on signals above 21 kHz. If the new Fs is to be set for 30 kHz, it should have no information on signals above 15 kHz. It is quite possible that the wave does contain information between 15 kHz and 21 kHz. If you don't get rid of this, alias distortion will result. To dump this, call up the EQ:Filters function and select a 5th order low pass filter. It should be set for no more than 15 kHz (15000 Hz - remember, k stands for "times 1000"). You may wish to set this for somewhat less, since the filter doesn't roll off infinitely fast. Once this is done, you may call up Sample Rate Transpose, and set it to 15000 Hz. Transposing to a higher Fs requires no pre-filtering operation.

Since the newly transposed wave will have a different number of sample points than the original, marker and loop offsets will no longer be "pointing" to the same instant in time. You can automatically re-align them by selecting the **Adjust Loops and Markers** check box.

This function can also be used to provide simple pitch changes in much the same way that keyboard samplers provide varying pitches. Unlike Resynthesis or the Pitch Shifter, this form of pitch shift will produce a corresponding time duration change (higher pitch leads to shorter sounds and vice versa). The process involves calculating a new Fs value with ReSample or Linear, and then bringing Fs back to where it started with Rate. For example, let's say you would like to lower the pitch of a wave by one semi-tone. The ratio of pitch change for one semi-tone is the 12th root of 2 (12 semi-tones in one octave, where an octave is a doubling of pitch). The 12th root of 2 is approximately 1.0595. If the present Fs is 10 kHz, transpose Fs up to 1.0595 times 10 kHz, or 10595 Hz. This will create a wave that sounds just like the original, but with a higher playback rate. If you now drop Fs back to 10 kHz using Rate instead of ReSample, only Fs is changed, not the wave data. Since the playback rate has dropped one semi-tone, the pitch of the wave will drop also. To raise the pitch, down sample with ReSample, and then raise Fs with Rate. Large pitch rises should be pre-filtered first. As a side note, you may discover on occasion that the value for Fs printed out in the Info dialog may not match exactly with what you specified for Fs. The reason for this is due to the afore mentioned problem with repeating numbers and fractions. Finally, always keep in mind that down sampled waves use fewer samples (less memory), while up sampled waves require more samples (greater memory).

At this point, you may be wondering why the process must require so much math. Well, it doesn't- we just like explaining how this stuff works. Instead of getting out a calculator and manually figuring out the required sample rate for a given pitch shift (like the example above), you can directly enter the desired shift and have Wrench do all of the work for you. In the New Rate box, enter the desired shift in cents. (100 cents equals one semitone, 200 cents equals one whole tone, etc.). You can enter values from -1200 cents (drop pitch one octave) to 1200 cents (raise pitch one octave). To activate **Quick Pitch**, select Quick Pitch from the Transpose Type group. You can still use the Adjust Loops and Markers and PreFilter items as you normally would. If you are raising pitch by more than a semitone or so and the sound contains a fair amount of higher harmonics, you will probably want to use the PreFilter option.

Unlike the other Sample Rate Transpose options, Quick Pitch does not always affect the entire wave. The portion of the wave pitch shifted depends on the setting of the Affect menu item, just like the other DSP functions available in Wrench. This is useful if you desire to shift perhaps one word in a sentence, or something similar. There is another unique use for this. Sometimes you desire to create a loop using just a single cycle of the wave. The problem is that unless the sampling rate is an integer multiple of the pitch of the wave, a single cycle won't contain a whole number of sample points. When this cycle is looped, you will hear a slight pitch shift (either sharp or flat depending on whether you truncated the cycle or added the "part-way" point). If you define the affect area as just this single cycle (perhaps using markers 0 and 1 with the corresponding Affect menu item choice), you can force the cycle to come back in tune with the rest of the wave.

Sample Rate Transpose also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Silence Insert

The Silence Insert function allows you to add space to a sample. This is useful if you need to isolate words in a narration, for example, or for special effects processing. You can specify how much silence you would like to add by using the amount text box. The units for this are set by the Type group. The choices are seconds, milliseconds, or sample points. The insertion point is specified by the buttons Sample Start, Sample End, Affect Start, Affect End, and Use Mouse. The first four of this group are for convenience and do as their names imply. The Use Mouse choice allows you to specify the insertion point by selecting with the mouse pointer, in the same manner as you would place a marker or specify a paste point. When the dialog box disappears after selecting Use Mouse, depress the mouse button over the waveform (either the main or overview areas). As you sweep the mouse, a tracking bar is drawn. Release the mouse button at the desired insertion point.

Statistics

The Statistics function does not alter waveform data, rather, it yields considerable information about the edit Affect area. It analyzes the segment and then reports the following information: The edit size, average value, DC offset value, RMS value, highest value, lowest value, crest ratio (ratio of peak amplitude to effective loudness), totals of possible positve and negative clipped points, and number of zero crossing in both the positive and negative directions. One nice thing about this function is that it reports the values using your native units. In other words, if you're using horizontal units of frames and vertical units of percent, then the edit size is reported in frames and the average, RMS, et.al. are reported as percents. Crest ratio is always reported using decibels and edit size is also shown using samples points.

Trim (Crop)

Trim will remove all areas outside of the edit Affect area. Conceptually, this is the exact opposite of the Delete function.

Unclip

Unclip is very unique function. The purpose of it is to "reclaim" waves which have been clipped, either due to overload while digitizing or excessive gain during editing. Mathematically, once a waveform has been clipped, that information is lost forever. It is, however, possible to take an educated guess concerning what used to be there, and that is precisely what Unclip does. In some cases, you will find Unclip to appear to work wonders, in other cases, you may wind up no better off than when you started. First and foremost, you should never rely on Unclip to fix the results of sloppy edits (such as excessive EQ boost, gain, or the like). These sorts of errors are best dealt with at the source and remedied with backups. Second, Unclip will work best on waveforms which have suffered only mild clipping, perhaps of a few decibels or less, and where any given clipped section spans just a few sample points. Sections which are very broad (ie, containing dozens of clipped points) cannot usually be reclaimed.

Using Unclip is generally a simple process, just select **Functions/Unclip**. Unclip needs no further input from you. First, it will analyze the waveform and estimate the amount of clipping in terms of how much "extra" gain seems to have been applied. Unclip will limit this to no more than 6 dB in an effort to avoid spurious transients. The signal will then be reduced by this gain and the clipped areas will be filled out. If the waveform was excessively clipped to begin with, the result of Unclip may still have some clipping artifacts.

Here are a few tips for making the most of Unclip. With very few exceptions, you should use the Maximize function prior to Unclip to ensure that the clipped portions will be at the proper level. It is generally a good idea to search the wave for badly clipped portions and perform a little touch-up on them before using Unclip. Look for sections which either span a large number of points (say, over 20 or so), or which tend to have very steep slopes (in other words, the portions on either side of the clipped area are almost vertical, containing just a few points). These portions can be rounded off using the Free Hand Draw pencil. Another item to consider is that even modest processing of a clipped signal can reduce the signals at the clipping point, thus making them appear to be unclipped. You can force these points back to the clipping level by applying .1 dB of gain to the signal (this will generally not affect the remainder of the wave). Once this is done, Unclip may be applied.

Effects (FX)

OK, here is where the fun begins. There is a menu for a bunch of neat sonic effects. This contains a series of "sample smoogers" which can be applied to your waveforms. The effects can be set for anything from subtle to sledgehammer severity. Many of the Effects have presets.

Amplitude Modulate (AM)

Amplitude Modulation is basically used for two things: creating tremolo or as a synthesis tool. The basic idea is to create a cyclic variation in volume. If this variation is fairly slow (a few seconds or so) and the total range of volume change is not great, the result is tremolo. Tremolo gives an undulating quality to a sound and makes it more dynamic. If the speed of volume variation is high (say, a few hundred Hertz), the result will be new harmonics, thus producing a change in timbre. This is basic AM synthesis.

The Amplitude Modulator has controls for **Modulation Speed**, **Modulation Depth**, and the **Initial Direction** of the sweep. **Modulation Speed** is adjustable from 10 seconds down to 1 millisecond (ie, 1000 Hz). Tremolo will generally use values from 10 seconds down to about .1 seconds, and AM synthesis will go from about .1 seconds to 1 millisecond.

Modulation Depth is adjustable from 1 to 100%. This represents the total percent change in volume. Small percentages produce subtle effects. Finally, **Initial Direction** can be set for Increase or Decrease. Increase starts the volume at the minimum level and works up to the normal level, while Decrease starts the volume at the normal level and works down to the minimum level. Please note that it is also possible to create AM effects using the Cross Multiply effect (see Cross Multiply for details).

Amplitude Modulate also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Chorus

Chorus imparts a thickened quality to a sound. This effect is based on combining the sound with versions of itself which have been delayed in time. This results in a multitude of images of the original sound, giving the impression of several items sounding at once (hence the name Chorus). There are five main settings for Chorus. They are **Delay Time**, **Feedback Percentage**, **Loudness Percentage**, **Modulation Speed** and **Modulation Depth**.

Delay Time is adjustable from 10 milliseconds to 60 milliseconds. **Feedback Percentage** is adjustable from 0 to 99% and indicates how much of the output signal is sent back through the Chorus to create a deeper effect. Normally, Chorusing uses delays in the 20 to 40 millisecond range with modest Depth and Feedback amounts (less than 30%, with suggested starting points in the 10 to 15% range). **Loudness Percentage** matches the affected signal to the original signal. Large values will tend to intensify the effect.

The Modulation section sets how the delay will be varied over time. **Modulation Speed** determines how long it takes to go through one "rise and fall" of the effect and is adjustable from .1 seconds to 50 seconds. Typical values are in the several second region. **Modulation Depth** indicates just how much of the initial Time Delay setting will be used for the sweep. A value of 100% means that the entire delay is used, which makes for very obvious sweeping. A low value such as 10% indicates that the sweep range will only be 10% of the delay. Note that extreme high settings of Delay Time and Modulation Depth can create a warbling quality to the sound, which is particularly noticeable when using faster Modulation Speed values. Such extremes do not represent traditional chorusing, but you may find these effects useful in special situations.

Be aware that applying high Feedback and Loudness percentages on waveforms that are close to full scale may create clipping distortion. If these parameters are required for the desired effect, the sound should be reduced using the Gain function prior to applying Chorus. Also, if you are performing spot edits, an easy way to soften the "edges" of the effect is to use the Edit Smooth capability on the edit Start and End areas. In general, the Smooth Time should be several times longer than the Delay Time.

Unlike many Chorusing units, it is possible to use some rather extreme values here. In fact, it is possible to create combination Chorus/Flange effects, or detuning effects. These are done by using large modulation Depth values, nearing 100%. In many contexts such effects are not very musical, but have uses in the more traditional "strange effects" area. An interesting variation on the "robotic voice" may be achieved with a Delay of about 25 milliseconds with Feedback and Loudness of about 85%. Set the Mod Speed to a few seconds and the Mod Depth to about 30%.

Chorus also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Convolution

Normal time domain convolution is useful for filtering, short echo/reverb, and special effects. If you're not familiar with the concept of convolution, think of it as the way a pulse is stretched and warped over time.The impulse, or convolvor, is simply a pattern which describes what will happen to the individual sample points. In this respect Convolution is conceptually similar to Impulse Modeling, although it is better suited here for short impulses.

To open the Convolution dialog, select **Effects/Convolution**. The display is dominated by a listing of existing (loaded) waves. You can specify the wave you wish to convolve with by clicking on it. You can also choose to use a sound file on disk. Generally, although long impulses are possible, they require insane amounts of processing time. It's best to keep the impulse under a few hundred points and use Impulse Modeling for longer ones. Like the Impulse Modeler, Convolution impulses are stored as normal sound files. This means that any sort of processing you can do on a sound, you can also do on an impulse. For the experimenter, much fun can be had using the freehand draw pencil on the impulse and then listening to the result.

For the advanced user, Convolution can also be used to implement FIR filters. The impluse response of the desired filter function is the convolvor.

Cross Multiply

This function multiplies one chunk of wave data by another. You can cross multiply two waves together, or a wave with a clip. Further, you can choose whether or not you'd like to use **signed multiplication** (ie, a four-quadrant multiplier for you technical types) vs. absolute (ie, a two-quadrant multiplier, which mimics a VCA).

The sonic result of multiplication is not at all like just adding two waves together. When two waves are multiplied, a complex frequency spectrum results. The product of the multiplication process contains attributes of its two "parents", but also has new characteristics as well.

The final adjustment on the Cross Multiply dialog is for scaling the results. If **Scale** is not set high enough, the resulting wave will be very low in amplitude, and thus very quiet (and probably noisy). If too great a factor is selected, the wave may distort. If you're trying to synthesize a new sound, you may actually like the distortion, so feel free to experiment. In general, if you're multiplying waves that are fairly low in amplitude to begin with, greater scale factors will be required.

Besides being an interesting effect, this function can be used to create tremolo effects. Tremolo is a variation of loudness over time (eg, modulating a volume pedal). To create tremolo, use unsigned multiplication with a low frequency wave as the multiplier (normally, just a few Hertz). A higher frequency multiplier will start to change the timbre of the wave through a form of amplitude modulation synthesis. The difference between this technique and using the Amplitude Modulation function is that here the modulator can be any waveshape. One possible source of modulator material is the Generate function (found under the File menu). You could, for example, create a 2 Hertz sine or triangle wave for the modulator. This wave could be further altered via Rectification, Integration, Gain scaling, Transfer Function, or DC Offset, just to name a few.

Differentiate

This function will create a new waveform which is the slope of the original wave. Yes, Wrench can now perform calculus on your sounds. Mathematically, the effect is similar to having a first order high pass filter set at a frequency higher than the highest frequency in the waveform. What this means is that sharp transitions in the wave are boosted, while slower changes are reduced. This results in a waveform which is brighter and lower in amplitude. Besides the sonic effect, Differentiation has use for people examining waveforms in non-musical areas. Finally, there is one other unique use of the function which may be of interest to those trying to compress sound files. By effectively recording only the differences in the waveform and reducing its level, you will generally find that compression utilities will create smaller files of differentiated waves than of normal waves. Differentiation is perfectly reversible using the Integrate function, so you can load the differentiated wave back into Wrench, Integrate it, and wind up with your normal wave. (Remember, you can think of Integrate as UnDifferentiate, or Differentiate as UnIntegrate.)

Echo

This is an effect familiar to almost everyone. It imparts a repeating quality to the sample, in much the same manner as shouting into a canyon reflects your words back to you. There are three settings for the Echo generator. They are the **Delay Time**, **Feedback Percentage**, and **Loudness Percentage**. The **Delay Time** specifies the spacing between each echo and is adjustable from .01 seconds (10 milliseconds) to 10 seconds. **Feedback Percentage** indicates how much of the output signal is routed back through the generator, thus causing echoes of the echoes. Large values will give many echoes, each being only moderately quieter than the echo preceding it. A small value will produce very few echoes, each significantly quieter than the echo before it. When Feedback is set to 0, only one echo will be heard. **Loudness Percentage** will set the overall level of the echoes relative to the original sound. A maximum setting sets the initial echo to be as loud as the original sound, with lower values making the echoes quieter.

Here are some application tips. For initial experimentation, try Delays in the .25 to 1 second area, with Feedback around 25 to 50%, and Loudness at about 50%. Tight slap echoes and doublings are achieved with very short delays of .01 to .05 seconds, Feedback of 0%, and Loudness in the 50 to 100% area. For a robotic character, try a short Delay of approximately .03 seconds with Feedback and Loudness of about 75%. In general, be wary of using large Feedback values along with large Loudness values, particularly on waveforms that are close to full scale. Such combinations can cause clipping of the signal. Finally, if you are performing spot edits, an easy way to meld the echoes into the non edited portion is to use the Edit Smooth capability on the edit End area. You may wish to extend the end point of the edit area by about one half of the Smoothing Time as well, in order to make sure that echoes extend far enough.

Echo also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Flange

Flanging imparts a characteristic swishing quality to a sound which is rather unmistakable. This effect is based on combining the sound with versions of itself which have been delayed slightly in time. This results in a series of phase cancellations, creating the effect of a sequence of notch filters (also known as a comb filter) which moves through the frequency spectrum. There are five main settings for the Flanger. They are **Delay Time**, **Feedback Percentage**, **Loudness Percentage**, **Modulation Speed** and **Modulation Depth**. There is also a check box for signal **Inversion**.

Delay Time is adjustable from 1 millisecond to 10 milliseconds. Shorter times tend to create more pointed effects. **Feedback Percentage** is adjustable from 0 to 99% and indicates how much of the output signal is sent back through the Flanger to create a deeper effect. Subtle effects are achieved with values below 10%, while sledgehammer effects can be produced with percentages in the 70 to 99% area. Selecting **Invert** can make the effect even more pronounced, and imparts a certain nasal, at times hollow, characteristic. **Loudness Percentage** matches the affected signal to the original signal. Large values will tend to intensify the effect.

The Modulation section sets how the delay will be varied over time. **Modulation Speed** determines how long it takes to go through one "rise and fall" of the effect and is adjustable from .1 seconds to 50 seconds. Typical values are in the several second region. **Modulation Depth** indicates just how much of the initial Time Delay setting will be used for the sweep. A value of 100% means that the entire delay is used, which makes for very obvious sweeping. A low value such as 10% indicates that the sweep range will only be 10% of the delay. Note that extreme high settings of Delay Time and Modulation Depth can create a warbling quality to the sound, which is particularly noticeable when using faster Modulation Speed values. Such extremes do not represent traditional flanging, but you may find these effects useful in special situations.

For a starting point, try "middle of the road" values such as 3 milliseconds of Delay with 50% Feedback, 75% Loudness, Mod Speed of 2 to 4 seconds, and Mod Depth of 40%. A very sharp quality can be produced with a Delay of 1.5 milliseconds, Feedback at 75%, Loudness at 90%, and Mod Depth at 50%. Note that the effect is more easily heard on sounds with strong high frequency content. Also, be aware that applying high Feedback and Loudness percentages on waveforms that are close to full scale may create clipping distortion. If these parameters are required for the desired effect, the sound should be reduced using the Gain function prior to applying Flange. Finally, if you are performing spot edits, an easy way to soften the "edges" of the effect is to use the Edit Smooth capability on the edit Start and End areas. In general, the Smooth Time should be several times longer than the Delay Time.

Flange also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Frequency Modulate (FM)

Frequency Modulation is basically used for two things: creating vibrato or as a synthesis tool. The basic idea is to create a cyclic variation in pitch. If this variation is fairly slow (a few seconds or so) and the total range of pitch change is not great, the result is vibrato. Vibrato gives an undulating quality to a sound and makes it more dynamic. If the speed of pitch variation is high (say, a few hundred Hertz), the result will be new harmonics, thus producing a change in timbre. This is basic FM synthesis.

The Frequency Modulator has controls for **Modulation Speed**, **Modulation Depth**, and the **Initial Direction** of the sweep. **Modulation Speed** is adjustable from 10 seconds down to 1 millisecond (ie, 1000 Hz). Vibrato will generally use values from 10 seconds down to about .1 seconds, and FM synthesis will go from about .1 seconds to 1 millisecond.

Modulation Depth is adjustable from .1 to 99%. This represents the total percent change in pitch. Small percentages produce subtle effects. Finally, **Initial Direction** can be set for Increase or Decrease. Increase produces an initial sharpening of pitch while Decrease produces an initial flattening of pitch.

Frequency Modulation also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Grunge

Normally, editors are used to clean up and polish audio. There are times, though, when sounds need an "antique finish". This is where the Grunge effect comes in. With it you can add background hiss to emulate an old tape player or noisey piece of electronics. You can add pops, clicks or crackle to simulate vinyl albums or bad connections. You can also recreate the sound of vintage samplers.

To call up the Grunge dialog, select **Effects/Grunge**. This dialog contains three major sections: **Hiss**, **Pops**, and **Bit Depth**. **Hiss Level** allows you to set the desired amount of hiss in the signal. 100% indicates full signal strength. This will normally drown out most audio. Typical values are in the 2 to 10% range. A setting of 0% produces no additional hiss. **Pop Level** controls the volume of clicks and pops. 100% indicates full volume while 0% inhibits the addition of pops and clicks. The actual level and duration of a given click or pop includes subtle variations in order to make them more realistic. **Pop Density** controls how frequent the pops and clicks are. To produce constant crackles, use a Density of 95% or greater. Densities in the 30 to 70% range are more typical of old LPs. The placement of the pops and clicks is not equally spaced; rather, random variations are added automatically to increase realism. **Bit Depth Reduction** indicates how many bits should be kept in the sound. CD quality is 16 bits. Removing bits adds noise and creates a somewhat "retro" quality. Many samplers from the mid-80's used 8 to 12 bits of resolution rather than 16. You can select 6, 8, 10, 12, or 14 bit results. If you don't want any bit reduction, select "None".

Grunge also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Harmony

Harmony does exactly what its name implies, it creates new sounds identical to the original except at a new pitch and mixes them in.

To call up the Harmony dialog, select **Effects/Harmony**. This dialog contains five sliders: a level slider and a pitch slider for each of the two new voices, and a level slider for the original wave. The **pitch** can be dropped down (flattened) by up to 2400 cents (2 octaves). The pitch can be raised (sharpened) by the same degree. **Volume levels** for each new voice can be independently adjusted from 0 to 100%. You can also adjust the level of the original signal from 0 to 100%. If you only need one new voice, set the volume level of the second voice to 0%.

Since new voices are being added, the overall effective volume will be somewhat louder so it is generally a good idea to run the levels below 100%. Depending on the source material, a range of 50 to 70% usually works well if one new voice is added. For two new voices, a range of 35 to 60% is typically in order.

Harmony also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Impulse Modeling

This goes by many names, including Acoustics Modeling, Ambience Modeling, and others. Basically, it's a great way to get reverb and special effects. To open Impulse Modeling, select **Effects/Impulse Modeling**.

The Impulse Modeling dialog contains a scrolling **wave list** at the top. You choose a wave from the list, or you load one from disk (**External File** option). We have decided to keep our impulses as simple WAV files rather than a proprietary format for greatest flexibility. This means that you can use anything that Wrench can load as an impulse file (think of the possibilities). The impulse file basically describes the time-domain transfer function, in other words, how a sound is "smeared" over time. This is perfect for reverb, but it's also useful for fancy filters and special effects. Think of the peaks you see in the impulse file as delayed copies of some original spike. An "identity" impulse would appear as a single spike at the very start of an otherwise silent file. Applying this gets you what you started with. If the single spike is simply moved say, 100 milliseconds into the silent file, then you will wind up delaying the wave by 100 milliseconds. A slap echo impulse would consist of just one spike some time after the start of the wave (along with the initial "identity" spike, of course). A constant echo would appear as a series of equally spaced spikes gradually decreasing in amplitude. Reverb consists of numerous spikes at varying distances, each representing some reflection off of a surface. Note that impulse modeling assumes that the "object of your emulation" is linear, thus distorting amps or amplitude/frequency modulation won't come out right (no vibrato, compression, pitch shifting, etc.). Of course, you can always use these sorts of impulse responses for special effects.

There are only a few controls. Each defines **optional processing** of the impulse:

Mix: Combine reverb signal with current wave info rather than replacing it. If the impulse doesn't have the initial spike, you generally want Mix. If it has an initial spike, you probably don't.

Reverse: Flip the impulse front to back. A wonderful effect (reverse reverb) when applied with Shift (see below).
Fade Out: Effectively shortens the impulse by speeding up the natural fading of the impulse.

Stereo Link: If you have a stereo source and a stereo impulse, selecting Link combines the two channels and uses that as the feed into the left/right impulses. This is not quite the same as mono reverb since left/right have different impulses. Without Link, each channel is processed independent of the other, thus a hard pan to one side will only get reverb in that one side.

Ignore starting milliseconds: This effectively removes the beginning of the impulse. Good for ignoring an initial spike for special effects or Mixing.

Ignore ending milliseconds: This effectively removes the end of the impulse and is also good for special effects, especially gated reverb.

Shift milliseconds: Where to place the reverb with respect to the signal you're editing. Negative values shift prior to the sound, positive values shift after the sound. The maximum for Ignore/Shift is the size of the impulse itself. If you specify something larger, Wrench will limit it for you.

Importing and creating your own impulses

Although we've included a few impulses for you, it's not difficult to obtain your own. There are three basic ways of getting impulses into Wrench:
1) Import them from other programs
2) Create them from the outside world (such as an impulse of your basement)
3) Generate them using Wrench's internal functions and effects

In all three cases the basic idea is to obtain a WAV file containing the impulse you're after. This is then loaded into the Impulse Modeler using the External File option. If you want to import an impulse from another program, you don't have to do anything special if the other program is nice enough to save their impulses as WAV files (or any other format Wrench recognizes for that matter). If the impulses are stored in a proprietary format, you have a little extra work to do.

At this point you're going to use the MasterImpulse.wav file found in the Wrench Sounds directory. MasterImpulse.wav is a two second long stereo file at 44.1 kHz. It is completely silent except for a single identity spike at the start of each channel. In each of the three cases above you'll feed this signal into the item of interest and then extract a WAV file which can be loaded into the Impulse Modeler. Since you'll want to keep MasterImpulse.wav as is to create other impulses, you'll want to create a copy of it and rename it. Here's what you want to do: Load MasterImpulse.wav into Wrench. If you're going to work with sounds at sample rates other than 44.1 kHz, you'll need to change the sample rate. Call up Functions/Sample Rate Transpose and enter the new sample rate. Select Rate Only as the style. Depending on the rate you've chosen, the file may now be somewhat longer or shorter in time. If you're target impulse is longer than what you have now, you'll need to fill the signal out. This can be accomplished with Functions/Insert Silence. Simply add as much silence as is needed to the end of the signal. Note that it is possible to make mono impulses as well by removing the right channel. Now that you have the impulse set up, save it under a new name (File/Save As). For the following examples, let's call this Master2.wav. At this point, the process diverges.

Case 1: Importing impulses using a proprietary format

Case 2: Creating impulses from the outside world

Case 3: Generating impulses from within Wrench

For **case one**, load Master2.wav into the other program just as you would any other sound file you were going to edit. Apply the other program's impulse function to Master2.wav using whatever impulse you're interested in. Once processed, Master2.wav will be a clone of the desired impulse. Save this under a new name if desired. This WAV file can now be used by Wrench's Impulse Modeler. That's all there is to it.

Case two is a little more involved. It does require some equipment and a little patience on your part. Basically, all you need to do is to play back a single spike through the item of interest (your favorite plate reverb, through a loudspeaker into your basement, etc.) and record the result. The result is then scaled and trimmed in Wrench like any other sound file, and is ready to use as an impulse. The major question is one of logistics: how is the impulse created and captured? There are many possibilities, all depending on whether AC power is available, the remoteness of the site or portability of the item, etc. For electronic devices in particular (plate or digital reverbs and similar devices), it may be best to play the MasterImpulse file into the device and then record the output with a second computer or DAT. For acoustical spaces, a microphone is placed at the desired "sweet spot" (the listener's position). The impulse will need to be fed into an amplifier and loudspeaker placed at a preferred position (the speaker's position). The resulting signal is picked up by the microphone and recorded, again either with a laptop computer or DAT. Of course, the microphone, loudspeaker, and amplifier will each add their own impulse coloration to the result. This arrangement can be a bit unwieldy, so there are options, notably in the generation of the excitation pulse. First, all that's needed is a good solid click for the excitation pulse. While it is convenient to use MasterImpulse for electronic devices, there's nothing that limits you to just this one signal. For example, many synthesizers and drum machines have a count-down or click sound which can be used. To avoid the hassles of an amplifier/loudspeaker setup for outdoor areas, a starter's pistol can be used if the microphone is at a considerable distance (this can scare the beejeezus out of other folks in the area, so please investigate any local noise ordinances and be considerate of others). Something a little less extreme is to clap a couple of blocks of smooth hardwood together. A handclap can even be used in a pinch. Each of these pulses is less than perfect, but they are pretty easy to produce. They will also impart some coloration to the impulse response. You may even prefer the coloration produced by some impulse sources over the theoretically perfect pulse. Remember, it's all subjective and your ears are the final judge. It is generally a good idea to create several impulse response "takes". This is to avoid background noise and interference, overloads, or insufficient recording level.

Once the recordings are finished, the raw responses will need to be cleaned up. Each take is loaded into Wrench in turn. If the signal shows clipping, insufficient level, or interference signals, it will probably have to be discarded. An ideal take will have an initial spike close to full amplitude, although the reverb tail may be considerably lower in level. The beginning spike should be trimmed right to the very start of the wave, removing anything before it (this is the identity spike mentioned earlier). After using vertical zoom-in, identify where the reverb tail falls off into the background noise. Chop off everything after this point. You may wish to apply further processing at this point, most notably using the Envelope Generator to smooth the fade out. You might also wish to combine several takes to create an average response. This can help reduce coloration and background noise. It is very important to make sure that the start spikes of each take are trimmed to the same exact location so that they overlay perfectly. Failure to do so can result in severe phase cancellation.

Case three is rather like case one, except that it requires some sense of experimentation on your part. Load Master2.wav into Wrench. Now, feed this wave into some series of Wrench effects. Some reverb or echo is a good place to start. Then try anything else, such as AM, FM, Transfer Function, EQ, you name it. Cut/Paste and simple Reverse edits can have amazing results. It can be difficult to get a good sense of what these edits will do when combined, especially when you first start experimenting, but then a little serendipity never hurts when you're in search of unique new sounds.

Some further ideas...

Above all, remember that the impulse files Wrench uses are just WAV files. Consequently, any sort of edit you can perform on a sound file in Wrench can also be used to alter an impulse. You are not limited to simple changes in envelope or filtering. This opens an entire world of creative possibilities.

For weird effects, try using something like a single piano note or guitar chord as the impulse. Try this on a spoken passage for a neat vocal! Some volume reduction may be required on the impulse first (for something like a piano note, reduce the impulse by about -20 to -30 dB using the Gain function). Interesting reverb-like effects may also be produced using sounds of a largely transient nature as the impulse. This includes sounds such as single drum or percussion strikes, cricket or bird chirps, and even thunder claps. Again, the amplitude will most likely have to be reduced to avoid saturation.

Here's how to do reverse reverb: Say your impulse is 1 second long. Select Reverse and set Shift to -1000 milliseconds (i.e., -1 second).

Here's how to keep your stereo mix, but spread the reverb behind it: Select Link and Mix. If the impulse has the identity starting spike, set Ignore Start to a few milliseconds. If the impulse doesn't have the start spike, don't bother with Ignore Start.

For those truly interested in realistic positioning of sources in an acoustic space, consider creating a binaural impulse response. This requires the use of a special binaural head in place of normal microphones. The impulse capturing process is unchanged from that outlined above, however, sources edited with a binaural impulse can produce a very unique experience when played back through appropriate earphones.

Integrate

This function will create a waveform which varies in accordance to the area of the original wave. For you calculus students, yes, this is "area under the curve", and it has a variety of uses in research areas, along side its sonic effect. The effect of Integration is to smooth out fast signal variation. It is akin to having a first order low pass filter which is set at an extremely low frequency, thus progressively reducing high frequency content. Waveforms of moderate amplitude and low frequency content, or even modest DC offsets can easily cause the Integrator to overload. The integer and floating point (24/96) versions of Wrench handle this differently. The floating point version simply creates a very large overscaled sound. The same approach would make the integer version clip, losing some information in the process. Instead of clipping, the signal is "wrapped around" the axis. Thus, very large positive values will appear as positively growing negative values and vice versa. By doing this, the Integrate function may be perfectly undone with the Differentiate function since no information about the wave is really lost. If you do not want the wrapping to take place, you should scale the waveform down using the Gain function first. The sonic effect of the wrapping can be quite interesting and useful in its own right. It tends to impart a bright, fuzzing quality, but unlike ordinary clipping distortion, wrapping doesn't tend to mask the original timbre of the sound. Instead, it imparts a metallic quality to the wave.

Generally, it is a good idea to use the DC Offset function before using Integrate. This prevents small DC offsets from causing runaway saturation problems.

Invert

This function flips a wave upside down. It does the same thing a phase reversal switch does on a mixing console. By itself, most people would never hear its effect. It is used primarily when combining sounds together to achieve different effects. Inverting an inverted wave will bring you back to where you started. Since this is a one shot sort of function, no dialog is used.

Pitch Shift

Pitch Shift is used to alter the pitch of a sound without changing its duration. While Resynthesis can be used for shifting pitch while keeping time constant, it is computationally expensive, requiring you to wait around for results. The Pitch Shift function allows you to create modest shifts of +/- 400 cents (4 half steps) in a comparatively short period of time. The exact time depends on the source material and the settings. In general, small shifts take less time than large shifts and Extended Bass will require somewhat more computation time than the normal setting.

This function is very easy to use. There is a single slider for setting the amount of **pitch shift**, in cents. Below the slider is a readout indicating what the resulting shift will be, which is updated as the slider is moved. There is also a checkbox for indicating whether or not the wave has **extended bass** content. Check this if there is reasonable content below about 70 Hertz. Typical sources with information in this area include kick drums, floor toms, the lowest octave of bass guitars, and the lower registers of organ, piano, and synth. Male voice does not fall into this category, so if you're editing something like a narration, don't check this box. Also, there's a **Stereo Link** checkbox. This is used to phase-lock both channels of a stereo wave. Without this locking, some stereo sources will exhibit phasing or delay artifacts after processing. For some stereo sources lcoking is not required (for example, two instruments, one panned hard right, the other panned hard left).

What about the sonic results? This varies with the settings and source material. In general, the end product varies from good to excellent. The best results are achieved with modest shifts (100 cents or less). As the shifts get larger, small timing and attack variances can be heard in some sources. The most difficult sources are those with a very strong rhythm and extended bass, such as a full production pop song. Songs of a more legato, flowing nature can take full 400 cent shifts with no noticeable side effects, as can most single instrument samples or voice-overs.

There are a few things to watch out for when using Pitch Shift. First, Wrench will attempt to get the precise shift that you request, but this is not always mathematically possible without introducing objectional clicks. Instead of doing that, Wrench tries to get as close as it can to your requested shift without producing clicks. For large shifts on complex sources, or with short samples of only a few seconds duration, there may be some variation in the timing, although it will usually be within a few percent of the original.

Finally, a few words about using **Extended Bass** are in order. First, if the source material contains information in the lower register, choosing not to use Extended Bass will most likely result in modulation of the wave, which is usually heard as a sort of strange rumbling or tremolo. If you select Extended Bass and the source doesn't contain low frequency information, this can degrade timing and shift accuracy. In some cases, selecting Extended Bass can give subjectively more pleasing results even if it's not required in theory.

Rectify Full Wave

This function produces the absolute value of the waveform. In other words, all negative portions of the wave are flipped up and become positive. The result is a wave which contains only positive amplitudes. This can be a very useful sonic effect, similar but not identical to overdrive distortion. All forms of Rectification can produce sizable DC offsets. This may create problems for certain samplers (typically, turn on/off thumps). In general, it is a good idea to use the DC Offset function after using Rectification.

Rectify Half Wave

This function removes all negative portions of a waveform, replacing them with silence. The result is a wave which contains only positive amplitudes intermixed with blank spaces. This can be a very useful sonic effect, similar but not identical to overdrive distortion. All forms of Rectification can produce sizable DC offsets. This may create problems for certain samplers (typically, turn on/off thumps). In general, it is a good idea to use the DC Offset function after using Rectification.

Resynthesize

Resynthesize gives you access to some powerful tools. To access them, select **Functions/Resynthesize**. It is useful for two things:

1) **Time stretch** (or contract) a wave without producing a pitch shift.
2) **Shift the pitch** of a wave without changing its time duration.

Both items are variations on a common theme, that of waveform resynthesis. The basic idea is that the source waveform is analyzed and broken down into fundamental components. These components can then be altered and reassembled (resynthesized) into a new waveform. The alterations performed on the components is what determines (in this case) whether you get pitch shifts or time alterations. Consequently, the setup for the initial waveform analysis is the same for both variations. This is why there is a single menu choice of Resynthesis instead of three separate choices. You have considerable control over the analysis parameters. These parameters include **Time Average**, **Frequency Resolution**, and **Signal Discrimination**. The analysis essentially chops the signal into small time chunks, and then determines the frequencies which exist in those chunks.

Time Average gives an indication of how much of the signal is used for the time chunk, over and above the required minimum. You can select Long, Normal, or Short. Long will produce more accurate results if you have a waveform which does not change very quickly. Very dynamic waveforms are better suited to the Normal and Short choices.

Frequency Resolution determines the number of different components (and their relative spacing) which the analyzer can see. Your choices are High, Normal, and Low. Generally, High will tend to give better results, but only for waveforms which do not change very quickly. Also, you should use High if you are trying to resynthesize a waveform with any low frequency content (below 100 Hz or so). The Low setting is useful for very dynamic waves which have little low frequency content. Low also calculates its results a little faster than Normal, which is in turn a little faster than High.

Signal Discrimination partially determines the accuracy of the analysis. It also plays a major role in determining how long it takes the analyzer to compute its results. Your choices are Fast, Normal, and Slow. Accuracy and computation speed are, unfortunately, inversely proportional (it's sort of a DSP version of "Haste makes waste"). Slow is the most accurate setting, but takes about 40% longer to compute than Normal, and about twice as long as Fast.

The items above must be set no matter which of the processes (**Pitch Shift** or **Time Stretch**) you choose. If you want **to perform a pitch shift**, adjust the **Pitch slider** to the desired amount. You can shift by up to +/-2400 cents (ie, sharp or flat by up to 2 octaves). 100 cents make up one semitone, and 12 semitones make up one octave. For example, if your note is presently at G and you wish to drop it to E (three semitones), set this to -300. On the other hand, if a sample was recorded out of tune and is about one half semitone flat, set this to +50. The setting of the Time Stretch slider will be ignored. Select the **Pitch Change** button under **Process Type**, and then select OK to start the resynthesis. Generally, small shifts will sound the most natural, but larger shifts can be used for effects.

For Time Stretching, ignore the Pitch slider and set the **Time slider** to the desired factor. This ranges from .25 to 4.0 times the present length. Note that this is not at all like clipping or appending chunks to the wave. The result will have the same general envelope as the original, but be either expanded or compressed in time. Small changes in time will be the most natural sounding. Extreme time changes can create bizarre alterations to the attack and release times of waves (but you may actually like this for certain sounds or for effects). One good use of this function is to adjust the length of a sound effect to fit within a certain time window. For example, you may have a sample of a sax which lasts for 2 seconds which you would like to use in a video. The segment for this sound is 2.5 seconds. This is 25% longer than what you have. You can stretch the sax to 2.5 seconds by setting the Time factor to 1.25 (2 seconds times 1.25 equals 2.5 seconds). To start the resynthesis, select the **Time Change** button under **Process Type**, and then select OK.

Please note that if you need only modest time alterations or pitch shifts on longer sounds (no more than +/-25%), consider using the Time Stretch and Pitch Shift items in Effects (FX) menu. For example, if you need to shoehorn a 32 second voice-over into a 30 second commercial, Time Stretch would normally be chosen over using Resynthesis. The power of Resynthesis is the range of shifts you can achieve.

Resynthesis also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Reverb

"Artificial ambience reconstruction" is the catch phrase here. The Reverb effect allows you to create the impression of a new acoustic space around a given sound. For example, you can make a tight, dry vocal sound as if the person was in the basement or a cave. Conceptually, reverb is very simple to understand. If you're standing in a room and suddenly clap your hands, a portion of the sound will go directly to your ears. This is referred to as the dry, or direct, sound. The remainder of the acoustic energy will radiate outward. When this energy hits a barrier, such as a wall, piece of furniture, or another human, a portion of it is absorbed and the remainder will fly off in new directions, only to repeat the process as it hits other barriers. How much is absorbed depends on the frequency of the energy and the material which the barrier is made of (as you might guess, large, flat, hard things are good at reflecting, and soft, mushy things are good at absorbing.) Of course, since sound travels only about 1.1 feet per millisecond, each bounce will probably take several milliseconds. There are, however, an uncountable number of reflections going on, so this gets very fuzzy in short order. Eventually, a portion of all of this reflected energy arrives back at your ears and your brain uses this information to estimate the size of the room and your position in it. Wrench attempts to simulate this process, giving you control over various parameters of the room. You can set the effect to range from subtle to overpowering. Reverb has the following settings:

Decay Time- This indicates how long it takes for the reverberated signal to die away (technically, to drop 60 dB from its initial value). This ranges from .2 to 20 seconds. The actual reverb time is a function of the program material and the setting of the Damping control. Sounds which are mostly mid and upper frequencies will have an apparently reduced time when used with moderate or having damping. Generally, large rooms have longer decay times.

PreDelay- This is adjustable from 0 to 200 milliseconds and is a way for you to delay the onset of reverberation. Moderate values in the 10 to 30 millisecond range tend to give the impression of a larger acoustic space and prevent the reverb from "stepping on" the original signal. Large values of predelay are useful for special effects.

Damping- Damping controls the amount of high frequency absorption in the acoustic space. High damping values in the mid-teens indicate that the room is absorptive, for example, as if it were carpeted, had heavy curtains and upholstered furniture. Rooms like this are said to be acoustically dead. Effectively, the decay time for the higher frequencies is shortened relative to lower frequencies. Low values of damping, say around 5 or so, show the opposite effect. This would correspond to a room with hardwood floors and little or no furniture, or perhaps a basement or bathroom. These rooms do little to absorb high frequencies and are said to be acoustically live. Extremely high or low values of damping tend to be unnatural sounding, but are useful for special effects.

Diffusion- The more complicated a room is, the more diffuse its reverb pattern will be. Generally, diffuse reverbs are described as lush or full. Geometrically simple rooms give rise to fewer complex reflections, resulting in a less diffuse, or simpler, sound. Diffusion is adjustable from 0 to 20. Note that rooms with high diffusion will tend to have the first reflections a little earlier in time, and Wrench simulates this effect.

Loudness- This controls the amount of reverb signal mixed in with the original signal. Be aware that reverb, especially with long decay times, can comprise a significant amount of energy, and thus it is quite likely that high percentages will cause clipping, especially if the source material is already near clipping. Generally, you can only use 100% loudness if the decay times are short. For moderate to long decay times, 100% loudness can usually be used if the source material is 3 to 6 dB below the clipping level.

Enclosure Type- You may choose one of three general types of acoustic spaces: Hall, Room, or Spring. Hall is the most complex space and will generally give the most lush and full reverb. On some material this can be too heavy. Room is somewhat simpler than Hall. You can think of it as a lighter form of reverb. Spring is a very simple reverb and is used mostly for special effects. It is not very smooth, but tends to simulate spaces with flutter echoes. Some might call it "boingy". It gets its name from the inexpensive mechanical reverbs that were very common in small mixers and guitar amplifiers. It is very important to note that it is perfectly acceptable to choose Hall with short decay times and Room with long decay times- you are not locked into using Hall with long decays and Room for short decays.

With all of the parameters available, you can create many different acoustic environments. It is very difficult to appropriately describe the sound of a given set of reverb values, so it is best if you grab something familiar, such as a short phrase, and try a few experiments in order to get a feel for what's going on. For starters, leave Damping, Diffusion and Loudness in the middle range, and select Hall. PreDelay should be left at 0. A good general reverb will be achieved with a Decay Time of around 2 seconds. A much tighter sound is possible with a setting of around .3 seconds, while a setting of 10 seconds will put you in a small cave. Once you're familiar with this, return to a moderate Decay of around 2 seconds and adjust Damping toward one of the extremes, listen to the result, and try the other extreme. Return Damping to a moderate level, and repeat the process in like manner with the Diffusion, PreDelay and Enclosure parameters. In order to hear the effect of the reverb, it is best if your sample is relatively short, but contains at least a few seconds of silence at the end. Without the silence, the reverb will be abruptly cut off, and you won't get a complete audition. (Make sure that this area of silence is included as part of the edit region!) If you need to pad a sound out with silence, an easy way to do this is to select Functions/Silence Insert. This will allow you to add a specific amount of space to your sound, leaving you with the original sound and a new silent segment for the reverb to wash into.

Cutting off the reverb tail can be useful at times. By purposely chopping off the ending when using long decay times, you can create "gated reverb". You can chop the end in one of three ways: 1) Bring the ending point of the edit region to a point after the ending of the sound but prior to the the decay time end point, 2) Select the area to be gated out as the edit area, and then select Functions/Silence, or 3) For maximum control, adjust the volume of the area to be gated out using the Envelope Generator.

Reverb also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

Reverse

This function is the back-masker's delight. Selecting **Functions/Reverse** will flop your wave front to back. If you preview it, it will play backwards. This can be used for some nice effects. Reversing the wave again will restore it to its original state. There is no dialog associated with this function, it's just a single shot.

Spectral Warp

This is a weird effect. Spectral Warp can do many crazy things to a waveform. It is particularly fun to use on voice. There are four slide controls: **Start Factor**, **End Factor**, **Start Bias** and **End Bias**. The **Factor sliders** set pitch shifts. For example, if they are set to .5 and 3 respectively, then the initial pitch is reduced by half and gradually increased until it gets to three times normal at the end of the wave. The **Factor Sweep** can be either Linear or Logarithmic.

The **Start** and **End Bias** produce timbre shifts and can also be swept in either a Linear or Logarithmic fashion. Large Bias values can create very strange sounding waves. Spectral Warp also includes the standard selection for internal ~~Presets~~, and the ability to Save or Load external presets. This is a unique function and we suggest that you check out the available presets on a simple voice sample for starters. The only down-side to this function is that it is rather intensive, often requiring minutes of computation for a few seconds of source material.

Time Stretch

Time Stretch is used to alter the length of a sound without changing its pitch. While it is possible to do this using Resynthesis, Time Stretch has been optimized for more modest shifts and ease of use, and as a result, operates much faster.

This function is very easy to use. It will produce a time alteration of up to +/- 25% (versus +/- 400% for Resynthesis). There is a single **time slider** for setting the amount of time compression or expansion. Below the slider is a readout indicating the present time length, and what the new time length will be, which is updated as the slider is moved. There is a checkbox for indicating whether or not the wave has **extended bass** content. Check this if there is reasonable content below about 70 Hertz. Typical sources with information in this area include kick drums, floor toms, the lowest octave of bass guitars, and the lower registers of organ, piano, and synth. Male voice does not fall into this category, so if you're editing something like a narration, don't check this box. Also, there's a **Stereo Link** checkbox. This is used to phase-lock both channels of a stereo wave. Without this locking, some stereo sources will exhibit phasing or delay artifacts after processing. For some stereo sources lcoking is not required (for example, a dialog between two people, one panned hard right, the other panned hard left).

Time Stretch is much faster than Resynthesis. The exact time depends on the source material and the settings. In general, small shifts take less time than long shifts, expansion generally takes longer than a similar compression setting, and Extended Bass will require somewhat more computation time than the normal setting.

The sonic results vary with the settings and source material. In general, the end product varies from good to excellent. The best results are achieved with modest shifts (10% or less). As the shifts get longer, small timing and attack variances can be heard in some sources. The most difficult sources are those with a very strong rhythm and extended bass, such as a full production pop song. (Generally, backing tracks for commercials are not as difficult since the announcer takes center stage in the mix). Songs of a more legato, flowing nature can take full 25% shifts with no noticeable side effects, as can most single instrument samples or voice-overs.

There are a few things to watch out for when using Time Stretch. First, Wrench will attempt to get the precise shift that you request, but this is not always mathematically possible without introducing objectional clicks. Instead of doing that, Wrench tries to get as close as it can to your requested shift without producing clicks. On short samples of a few seconds or less, shifts of more than 10% may come back on the shy side. Also, depending on the source material, very large shifts of 25% or so may be off by a few percent (again, most likely on a complex full mix).

Finally, a few words about using **Extended Bass** are in order. First, if the source material contains information in the lower register, choosing not to use Extended Bass will most likely result in modulation of the wave, which is usually heard as a sort of strange rumbling or tremolo. If you select Extended Bass and the source doesn't contain low frequency information, this can degrade timing and shift accuracy. In some cases, selecting Extended Bass can give subjectively more pleasing results even if it's not required in theory. Also, please note that Time Stretch can alter your perception of vibrato and tremolo. In general, time compression will make vibrato and tremolo more noticeable (and possibly objectionable) since it will naturally speed up the vibrato/tremolo frequency. On the other hand, time expansion will sometimes make vibrato and tremolo less noticeable, although a long stretch can make it more obvious since the sound is "hanging around" longer.

Transfer Function

A transfer function details how a system alters an input signal to produce an output signal. It can be expressed as an equation, or more directly, drawn as an X-Y graph. A typical graph will have the input signal running along the horizontal (X) and the output signal along the vertical (Y). Every input value has a corresponding output value. If the system is perfect and produces no alterations or distortions, there will be a direct proportion between input and output, and the result will be a straight line. If the slope of this line is other than unity, the system produces a gain (or loss). Any deviation from a straight line will change the timbre of the input wave. The more drastic the shape, the more the timbre will be altered. This explanation is somewhat simplified, but is fine for our purposes.

You can draw just about any transfer function you wish for your wave. The majority of this dialog is the **X-Y drawing area**. It is large so that you can draw with reasonable accuracy. The horizontal axis represents the input signal, with positive values to the right and negative values to the left. The vertical axis represents the output signal. Positive outputs are in the upper half while negative outputs are on the lower half. Drawing the transfer function is done in same manner as using the Free Hand Draw mode. Select and hold the left mouse button to draw. Left to right motion draws, while right to left motion erases.

Many times, you'd like the transfer characteristic to be symmetrical about zero (ie, equal effect for positive as well as negative signals). To avoid drawing both parts of the curve, draw just the positive part (the upper half of the display) and select the **Mirror** checkbox. Mirror will automatically calculate the inverse of the positive section for you. To initiate the calculation, select OK.

The sonic effects of a given transfer function can be hard to predict. Repeated use of this function will allow you to come up with some generalizations, though, and the presets are a good place to start. New curves are best left to your own experimentation. Don't be surprised if the same curve turns one wave into mush, yet turns another wave into something very interesting. A little serendipity never hurts!

Transfer Function also includes the standard selection for internal Presets, and the ability to Save or Load external presets.

General Reference

User Configuration on Startup

Wrench gives you control over how it is to start, in terms of editor attributes, main settings and so on. Wrench looks for three default files on start up. These files are: **wrench.config** (the default configuration), **wrench.macro** (the default macro file/function key assignment), and **wrench.macroinit** (the initial macro script to run). wrench.config and wrench.macro are created via **Setup/Save Config** and **Setup/Save Macros**, respectively. wrench.macroinit is a standard Enable script which you write.

Utilities

Earlier versions of Sample Wrench included a utility named CVPT. You might hear of it from a long-time Sample Wrench user. With the increased processor speed, RAM, and hard drive space available on modern systems, it is of limited use now and has been removed. The original description is included below for nostalgic entertainment.

CVPT

CVPT is a combination Converter, Viewer, Player and Transfer utility. It is, in essence, a subset of Sample Wrench. With CVPT, you can load sound samples and then save them using a different format, inspect sound samples, transfer sound samples back and forth between the computer and a MIDI sampler, preview a sample using the computer's internal voices, and remotely trigger a MIDI device.

CVPT opens a single editor window. This is a more or less standard editor window as found in Sample Wrench. Using the border buttons and sliders, you can zoom in and out of the wave, and set your desired position.

CVPT has just a few menus. The first allows you to save and load samples, set the file format type, select the desired sampler protocol and call up the Play Prefs dialog. You can also initiate a MIDI send or receive from this menu. The second menu allows you to set desired options such as horizontal and vertical axis calibration units, Overviews, XY Readout, ColorPoint, and the like. All of these items work exactly like their Sample Wrench counterparts.

If you understand how to navigate through Sample Wrench, then CVPT will pose no problems. CVPT was created for those times when you don't feel like opening up Sample Wrench because you don't need to use any of its powerful editing features, but just need to do a few format conversions, double check to see if a wave is the one you think it is, and so on. CVPT is much smaller than Sample Wrench, and so will make fewer memory demands on your system. Also, it will co-exist on the desktop with other programs. You will find CVPT to be very handy at a variety of times. One good example is if you plan on transferring a large bank of edited sounds from a sampler to the computer, or vice versa.

System Wide Background Menus (always available)

These menus are active when the background area is active. The direct indication of this is the mouse pointer. When the background area is active the mouse pointer will look like an arrow with tail feathers. These actions and attributes affect the entire system and are not keyed to a particular editor. Menus and items are given in order of appearance, left to right and top to bottom.

<u>File</u>

New Editor	Opens virtual editor window. Up to 99 are available.
Exit	Quits Sample Wrench. Closes all open editors. All dialogs must be answered before quitting.

<u>Setup</u>

The Affect group determines which segment of a wave will be affected by edits.

Affect All	Entire wave is affected by edits.
Affect In View	The area presently in view is affected by edits.
Affect Markers 0,1	The area between Markers 0 and 1 is affected by edits.
Affect Mouse	The area defined with the mouse is affected by edits. (The Affect area is shown in reverse highlight).
Edit Left	If selected, editing will affect the left channel of stereo waves. This should be selected if the wave is mono.
Edit Right	If selected, editing will affect the right channel of stereo waves.
Edit and Play	Automatically triggers playback after an edit.
Abortable	Allows functions to be halted in mid-stream.
Auto Zoom Out	If selected, edits which change the size of the wave will force a redraw such that the entire wave is visible.
Backups	Sets whether or not wave backups are available (hence, Undo).
Smoothing	Allows edit areas to be crossfaded, producing a smoother transition.
Configuration Files:	
Save Config	Saves current operating environment as a file.
Load Config	Loads configuration file and resets operating environment.
Macros:	
Save Macros	Saves current Enable macros as a file.
Load Macros	Loads an Enable macro file.
Assign Macros	Allows editing of the 11 Enable macro function keys.
Auto-Record Macro	Generates a macro from user actions.
File Paths:	
Sounds Path	Sets the default path for sound files.
Presets Path	Sets the default path for preset files.
Toolbars	Opens Tool windows. Five choices: Clips, Functions, FX, Loops+Markers, Views.

Format

Sets the format type for waveform saves to disk. Choose from:

WAV8	8 bit Windows format
WAV16	16 bit Windows format
WAV24	24 bit 3-byte-packed Windows format
WAV32	32 bit floating point Windows format
8SVX	Mono 8 bit Amiga format
AIFF16	16 bit Macintosh and Amiga format
AIFF24	24 bit Macintosh and Amiga format
AU	Sun's .au format, Next's .snd format
RAW8S	Raw 8 bit signed
RAW8U	Raw 8 bit unsigned
RAW16M	Raw 16 bit Motorola form
RAW16I	Raw 16 bit Intel form
RAW a-Law	Raw a-law encoded
RAW u-Law	Raw u-law encoded
RealAudio	Progressive Network's formats
SoundDesigner 1	Mono 16 bit Macintosh format
Studio 16 Version 3	Mono 16 bit Amiga format
VOC	Creative Labs Sound Blaster format
WAV Options	Check to save the desired WAV format optional chunks
smpl	Loops, rootnote, finetune
inst	Full keymap, fine tune
cue	Markers
INAM	Name

Sampler

Sets the MIDI sampler communication protocol for data transfers. Choose from:

SMDI	SCSI sample dump
SDS 12 Bit	Sample Dump Standard for 12 bit devices.
SDS 16 Bit	Sample Dump Standard for 16 bit devices.
Prophet 2000	Sequential Circuits P2000 or P2002
Ensoniq EPS	
Ensoniq EPS16+	
Ensoniq ASR10	
Akai S612	
Korg DSS1	
Korg DSM1	

Virtual Editor Menus

Each of the virtual editors has the same set of menus, in addition to the basic ones just noted.

<u>**File (extended version)**</u>

New Editor	Opens virtual editor window. Up to 99 are available.
New	Nulls present editor contents and all referenced (non-copied) clips.
Open	Transfers disk based sound file to this editor.
Save	Transfers present editor contents to disk, using present path and file name.
Save As	Same as Save, above, but with newly specified path and filename.
Send	Initiates a transfer from Wrench to a MIDI sampler.
Receive	Initiates a transfer from a MIDI sampler to Wrench.
Record Prefs	Set record preferences.
Record	Initiate a record session.
Generate	Create simple waves.
Play Prefs	Set playback preferences.
Play	Plays the entire wave.
Play Affect	Plays the present edit Affect area.
Stop Play	Halts all playback.
Properties	Present general information on present wave, including marker and loop status, path, sampling rate, and size.
Name	Specify a new name for the wave. This name appears in the editor's title bar.
File List	Opens recently used files.
Close	Shuts down this editor.
Exit	Quits Sample Wrench. Closes all open editors. All dialogs must be answered before quitting.

<u>**Edit**</u>

Undo	Undo last edit action. Available only if Setup/Backups is at least one.
Redo	Redo last Undo action. Available only if Setup/Backups is at least one.
Undo History	Shows recent edits. Jump forward or backward in list. Available only if Setup/Backups is greater than one.
Cut	Remove present Affect area from wave and place it in the system clipboard.
Copy	Copy present Affect area from wave and place it in the system clipboard.
Paste	Replace Affect area in wave with contents of system clipboard.
Delete	Cut out the present edit Affect section.
Mute	Silences the present edit Affect area.
Trim	Removes all sections outside of the present edit Affect area.

Multi-Clips Submenu

Clip	Create a clip with either mouse or markers, and place in clipboard.
Copy	Copy (de-reference) active clip.
Cut	Remove present clip from wave.
Paste	Place active clip in wave, pushing wave data aside.
Replace	Place active clip in wave, overwriting wave data.
Edit	Set new active clip, confirm clip values, load, save, delete, or play clips.
Erase	Null active clip.
Erase All	Null all used clips.
Play	Play the active clip.

Note: One multi-clipboard is shared among the virtual editors. This is how data is passed and shared among them. It is not the same as the system (Windows) clipboard.

<u>View</u>

Show Full	Shows the entire wave.
Zoom In Horiz	Zooms into the wave 2X horizontally.
Zoom In Vert	Zooms into the wave 2X vertically.
Zoom Out Horiz	Pulls out from the wave 2X horizontally.
Zoom Out Vert	Pulls out from the wave 2X vertically.
Horizontal Units	Sets horizontal calibration units. 13 sub-item choices; Total seconds (default), minutes and seconds, sample Words, frames per second in either total frames or hours:minutes:seconds:frames format using 24, 25, 30, or 30 drop frame, total beats, and measures plus beats.
Vertical Units	Sets vertical calibration units. Five sub-item choices, Percent (default), sample Value, dB, -dB, and -dB RMS.
Box Outline	Draws a box around the active view area for quick identification.
Color Point	Lights individual sample points with a complimentary color.
Overview	Draws a reduced version of the complete wave directly above the normal display.
X-Y Read Out	Mouse movements are reported in the appropriate calibration units.
Edit Status	Creates an info line at the bottom of the editor which indicates the present edit area and type.
Set Colors	Allows you to define the colors used for the waveform drawing.
Set Font	Allows you to specify the font used for editor labels.
Set Offset/Tempo	Allows the creation of a time offset to be added to the displayed horizontal axis values, and to set tempo values for the horizontal axis.
Set View	Sets one of ten alternate views of a waveform.
Get View	Recalls one of ten alternate views of a waveform.

<u>Mode</u>

Normal	Default operation mode. All functions available. Direct grabbing of markers and loop points with the mouse available. Mouse pointer is the default pointer.
Zoom Box	Allows fast zoom in operation via the mouse. All functions available. Mouse pointer looks like a magnifying glass.
Free Hand Draw	Allows direct wave drawing with the mouse. All functions available. Mouse pointer looks like a pencil.
Scrub	Allows playback using the mouse pointer. Mouse pointer looks like a hand.

<u>Functions</u>

Clone Wave	Make a copy of an existing editor/wave.
Combine Samples	Join present wave with a wave from another editor, with offset.
Cut/Keep List	Use Markers to save or remove multiple waveform segments.
FFT Spectrum Analysis	Calculate and display 2-D and pseudo-3D spectral analysis graphs.
Mono/Stereo	Turn a stereo wave into mono and vice versa.
Sample Rate Transpose	Change sampling frequency to a new one.
Statistics	Gives details about Affect area, including peaks, RMS level, zero crossings, etc.
Equalization:	
Bass/Treble	Bass and Treble shelving EQ.

Filters	First, Second and Fifth order high and low pass.
GraFreq	Five band graphic with adjustable tuning frequencies (semi-parametric).
Parametric	Two independent bands of parametric EQ.
Level Control:	
Compression	Standard compressor with threshold level, compression ratio, and attack and release time settings. Can also be used for expansion.
Envelope Generator	Draw arbitrary envelope with the mouse.
Gain	Apply gain or loss to the wave for scaling purposes.
Maximize	Make wave as large as possible.
Normalize	Scale wave to desired peak level.
Noise Gate	Remove low level segments.
Looping and Keymaps:	
Crossfade Looping	Calls up Crossfade Looping dialog. Automated looping functions.
Interactive Loop Window	Visual looping aid.
Key Map (simple)	Set root key and playback range.
MIDI Keyboard (with map)	Opens 128 key clavier for triggering MIDI devices, the sound card, or to set the key map graphically.
DC Offset	Add DC to wave.
Reduce Noise	Reduces constant background noises such as hum.
Remove Clicks and Pops	Removes clicks, pops, and other transient noises.
Replicate	Make repeated copies of wave data.
Silence Insert	Add a silent chunk to the wave.
Unclip	Smooth out clipping artifacts.

<u>Effects (FX)</u>

AM	Amplitude Modulate (tremolo or synthesis).
FM	Frequency Modulate (vibrato or synthesis).
Convolution	An impulse based effect good for special effects, filtering, and simple reverb.
Cross Multiply	An effect which combines two waves yielding a sound with attributes of the original, plus its own unique characteristics.
Chorus	"Thicken" the sound.
Flange	Classic "jet plane sound" effect.
Echo	Make controlled repeats.
Impulse Modeling	Acoustics, ambience, and device modeling.
Reverb	Artificial ambience reconstruction.
Grunge	Add noise, pops, clicks, and bit reduction.
Differentiate	Generate slope of waveform.
Integrate	Generate "area under the curve".
Invert	Flip wave upside down.
Reverse	Flop wave back to front.
Rectify Full Wave	Flip negative portions to positive.
Rectify Half Wave	Replace negative portions with silence.
Harmony	Create second and third voices at new pitches.
Pitch Shift	Alter pitch without changing length.
Resynthesize	Make drastic time or pitch shifts.
Spectral Warp	Shift pitch and timbre over time.
Time Stretch	Alter length without changing pitch.
Transfer Function	Draw arbitrary input/output function with the mouse.

<u>Loops+Markers</u>

Auto-Locate	Set parameters for auto location in Marker Set and Loop Set.

Loop Set Create new loops or edit existing ones.
Loop Show Draw loops on wave. Three sub-items are available. All draws all loops (except those explicitly set as "No Show" loops in the Set Loop dialog). None will not draw any loops, making an uncluttered display. Sus/Rel Only draws just the sustain loop (at bottom) and the release loop (at top).
Loop View Change the wave view so that the desired loop will be within the displayed area of the editor window.
Marker Set Create new markers or edit existing ones.
Markers Show Draw markers on wave. Default is active.
Marker View Change the wave view so that the desired marker will be within the displayed area of the editor window.

Menu Command Key Shortcuts

All menu shortcuts are accomplished by first pressing the ALT key, and then the keys as listed below.

Setup menu - P

Affect All	A
Affect In View	V
Affect Markers 0,1	0
Affect Mouse	M
Edit Left	L
Edit Right	R
Edit and Play	E
Abortable	T
Auto Zoom Out	Z
Backups	B
Smoothing	S
Save Config	F-S
Load Config	F-L
Save Macros	C-S
Load Macros	C-L
Assign Macros	C-A
Auto-Record Macro	C-R
Sounds Path	P-S
Presets Path	P-P
Toolbars Functions	O-F
Toolbars Effects	O-E
Toolbars Loops+Markers	O-L
Toolbars Multi-Clips	O-C
Toolbars Views	O-V

Format menu - T

WAV8	V
WAV16	W
WAV32	2
8SVX	X
AIFF16	F
AIFF24	4

AU	N
RAW 8S	S
RAW 8U	U
RAW 16M	M
RAW 16I	I
RAW a-Law	A
RAW u-Law	L
RealAudio	R
SoundDesigner	D
Studio 16 V3	3
VOC	O
smpl	V-S
inst	V-I
cue	V-C
INAM	V-N

Sampler menu - S

SMDI	M
SDS 12 Bit	D
SDS 16 Bit	S
Prophet 2000	P
Ensoniq EPS	E
Ensoniq EPS16+	Q
Ensoniq ASR10	A
Akai S612	I
Korg DSS1	K
Korg DSM1	O

Window menu - W

Cascade	C
Tile	T
Arrange Icons	I
Close All	A

Help menu - H

Contents	C
FAQs	F
How To's	H
Tip of Day	T
Tips at Startup	S
About Wrench	A

Virtual Editor Menus

File (extended version) - F

New Editor	W
New	N

Open	O
Save	S
Save As	A
Send	D
Receive	R
Record Prefs	F
Record	E
Generate	G
Play Prefs	Y
Play	P
Play Affect	L
Stop Play	T
Properties	I
Name	M
Close	C
File List	1 through 5
Exit	X

Edit menu - E

Undo	U
Redo	E
Undo History	H
Cut	T
Copy	C
Paste	P
Delete	D
Mute	M
Trim	R
Multi-Clips	
Clip	L-L
Copy	L-O
Cut	L-U
Paste	L-S
Replace	L-R
Edit	L-D
Erase	L-E
Erase All	L-A
Play	L-Y

View menu - V

Show Full	S
Zoom In Horiz	Z
Zoom In Vert	I
Zoom Out Horiz	H
Zoom Out Vert	V
Horizontal	
Seconds	A-S
Min:Secs	A-M
Words	A-W
24	A-F
25	A-2

	30 Drop	A-D
	30	A-3
	24 HMSF	A-H
	25 HMSF	A-5
	30 Drop HMSF	A-R
	30 HMSF	A-0
	Beats	A-B
	Meas+Beats	A-E
Vertical		
	Percent	R-P
	Value	R-V
	dB	R-D
	-dB	R-B
	-dB RMS	R-R
Box Outline		B
Color Point		C
Overview		O
X-Y Read Out		X
Edit Status		E
Set Colors		L
Set Font		F
Set Offset/Tempo		T
Set View		W (1 through 9, and 0 for subitems)
Get View		G (1 through 9, and 0 for subitems)

Mode menu - M

Normal	N
Zoom Box	Z
Free Hand Draw	F
Scrub	S

Functions menu - U

Clone Wave	W
Combine Samples	C
Cut/Keep List	I
FFT Spectrum Analysis	F
Mono/Stereo	M
Sample Rate Transpose	T
Statistics	A
Equalization	
Bass/Trebl e	Q-B
Filters	Q-F
GraFreq	Q-G
Parametric	Q-P
Level Control	
Compression	L-C
Envelope Gen	L-E
Gain	L-G
Maximize	L-M
Normalize	L-N

Noise Gate	L-A
Looping and Keymaps	
Crossfade Looping	K-C
Interactive Loop Window	K-I
Key Map (simple)	K-K
MIDI Keyboard (with map)	K-M
DC Offset	D
Reduce Noise	N
Remove Clicks and Pops	P
Replicate	R
Silence Insert	S
Unclip	U

Effects (FX) menu - E

AM	A
FM	Q
Convolution	L
Cross Multiply	M
Chorus	C
Flange	F
Echo	E
Impulse Modeling	M
Reverb	R
Grunge	G
Differentiate	D
Integrate	I
Invert	N
Reverse	V
Rectify Full Wave	U
Rectify Half Wave	H
Harmony	O
Pitch Shift	P
Resynthesize	Y
Spectral Warp	W
Time Stretch	S
Transfer Function	T

Loops+Markers menu - L

Auto-Locate	A
Loop Set	L
Loop Show	
All	O-A
None	O-N
Sus/Rel Only	O-R
Loop View	
Sustain	P-S
Sustain Start	P-U
Sustain End	P-T
Release	P-R
Release Start	P-E
Release End	P-L

Marker Set	M
Markers Show	S
Marker View	V (0 through 9 for subitems)

Non-Menu Keyboard Shortcuts

Window/Cascade:	Shift + F5
Window/Tile:	Shift + F4
Help:	F1
New Editor:	Ctrl + E
New:	Ctrl + N
Open:	Ctrl + O
Save:	Ctrl + S
Save As:	Ctrl + A
Undo:	Ctrl + Z
Redo:	Ctrl + Y
Undo History:	Ctrl + H
Cut:	Ctrl + X
Copy:	Ctrl + C
Paste:	Ctrl + V
Delete:	Delete
Mute:	Ctrl + M
Trim:	Ctrl + T
Play Affect:	Space Bar
Play:	Ctrl + Space Bar
Stop Play:	Shift + Space Bar
Scroll By Line:	Cursor Keys
Scroll By Page:	Shift + Cursor Keys
Show Full:	Home
Zoom In Horiz:	Ctrl + Left Cursor
Zoom In Vert:	Ctrl + Up Cursor
Zoom Out Horiz:	Ctrl + Right Cursor
Zoom Out Vert:	Ctrl + Down Cursor

"On the fly" Marker Create: Hold shift and click left mouse button at desired point on the wave. Next available ID is used.

Marker Set/Create:	Shift + Ctrl + 0 through 9 (no dialog)
Marker View:	Ctrl + 0 through 9
Set View:	Shift + Alt + 0 through 9
Get View:	Alt + 0 through 9
Enable macros:	Function keys 2 through 12

MIDI Keyboard Samplers

Sampler Communication

Sample Wrench allows you to send wave data back and forth between the computer and a variety of samplers via MIDI. You can load data from one sampler into Wrench, and then send it back to an entirely different sampler, if desired. In this way, you can use Wrench as a sort of "translator" between incompatible samplers.

In order to send data to and from various manufacturer's samplers, Wrench utilizes sampler drivers. A sampler driver contains the information needed by Wrench to successfully communicate with a given sampler. Before a MIDI data transfer is initiated, the appropriate sampler driver must be chosen from the Sampler menu (on the Amiga, select Sampler under the system-wide (background) menu MIDI). If you need to transfer a sound file to other samplers, their libraries will have to be chosen in turn. Failure to set the correct sampler driver may result in error messages or erroneous data. In extreme cases, an incorrect driver may "hang" the system. Be aware that you can always abort a MIDI send or receive in progress (by clicking on the Abort window).

There are two ways to configure sampler communication. The first way is static and the second is dynamic. Static is generally preferred if you only have one sampler, while dynamic is more convenient if you have multiple samplers. In the static form, you select a desired sampler from the **Sampler** menu. When you start a sample dump (by either hitting the S or R buttons in the toolbar, or selecting File/Send, File/Receive), a transfer dialog will pop up. The title at the top of the dialog tells you what kind of operation you've selected, and which sampler is selected. For example, if your current sampler driver is for SDS 16, and you click on the S button, then a dialog will pop up with a title like "16 bit Sample Dump Standard". Each sampler driver has its own specific dump dialog, which may be slightly different from the others, depending upon the parameters your sampler needs you to supply. For example, an Ensoniq EPS needs a layer number, whereas the Prophet 2000 doesn't even have layers.

To use the dynamic form, make sure that **Dynamic Dialog** is checked under the **Sampler** menu. In this form, the same style dialog will always be used. The area on the left side is sampler-specific and depends on the **Protocol** chosen. You can think of it as the static dialogs described above embedded in the left side. As you change the protocol, the left side items change. The dynamic dialog also allows you to create **Presets**, just like you can for Wrench's Functions and Effects. If you have a studio with several samplers, presets are very convenient. Each preset can have a descriptive name and includes the chosen protocol, MIDI channel, etc. For example, if you have an Ensoniq ASR on MIDI channel 2 and a Peavey DPM-SP using SMDI with a SCSI ID of 3, you can create a preset for each of them. Calling up the DPM-SP preset automatically sets the protocol to SMDI, the SCSI ID to 3, and so forth. You won't have to remember how every sampler is patched together in the system.

During a send or receive, the mouse pointer will change appearance to indicate the appropriate Send or Receive mode. On Windows, the progress bar will track the percent completion of the sound transfer.

All drivers require a "handshaking" connection for optimum performance and efficiency. This means that the sampler's MIDI Out should be connected to the computer's MIDI In, while the computer's MIDI Out should be connected to the sampler's MIDI In. These connections should be made before Sample Wrench is used. Also, make sure that the sampler is turned on and in the appropriate mode (if required). Generally, you should not make or break electrical connections to your computer while it is on.

The way in which various samplers send and receive data varies considerably. You don't have to worry about this variation though, as Wrench takes care of most everything. There are a few common error messages which you may receive, and for the most part, the fixes are rather straight forward.

Error Message	Meaning and fix
Can't find driver	The required sampler driver could not be found. Try reinstalling Wrench, paying attention to the sampler driver's section.
Can't open driver	System may be out of memory, or the driver may have somehow gotten corrupted. For the first case, free-up system memory by closing down unneeded programs. In the second case, reinstall the sampler drivers from the backup copy.
Driver in use	Another program is currently using the driver. Either wait for it to finish or terminate it, in order to free up the driver.
Out of Memory	Not enough memory for the incoming sample data. Free-up memory by closing down unneeded programs, or deleting unused waves in the other editors.

A listing of the current drivers follows. Each section lists the samplers supported and any known bugs or limitations of the samplers. Please note that manufacturers do update the operating systems of their samplers. Older operating systems may have bugs not listed here. It is generally advisable to get the most recent operating system update for your sampler.

Sampler Drivers

Sample Wrench works with a variety of different manufacturers' samplers. You can send and receive data to and from different devices with very little work on your part. All that you need to do before sending or receiving MIDI data is to specify which sampler is being used. Different manufacturers have set different communication protocols for their products. Before Wrench can talk with a sampler, it needs to understand the sampler's language. Wrench gets the needed information from its sampler drivers.

Choosing a sampler is very straight forward, just select **Sampler**. All of the available samplers will be shown on the menu list. Choose the sampler you want by selecting it from the menu. By default, Wrench uses the MMA Sample Dump Standard version. If the Sampler menu's **Dynamic Dialog** item is checked, then the Send and Receive dialogs can be changed to any available protocol on the fly, and using the Sampler menu choices is not required. If you have several different samplers, you will probably prefer Dynamic Dialog.

MIDI Interconnections

No matter which sampler you are talking to, the communication process is similar. First of all, connection between the computer and the sampler must be two way. The computer's MIDI Out must be connected to the sampler's MIDI In and the sampler's MIDI Out must be connected to the computer's MIDI In. This is referred to as a handshaking connection and makes the data transfer faster and less prone to error. MIDI cables should be kept relatively short to prevent signal degradation.

Receiving a Sample Dump

To receive a sample, click on the **R** button in an editor's toolbar, or choose **Receive** from the **File** menu. If the editor already contains a wave, you will be asked if you want to erase the present wave in order to make room for the new one. A small MIDI window or dialog will open in the main Wrench window. The exact look of the MIDI window will depend on the sampler model chosen (ie, the driver selected from the Sampler menu or the protocol chosen if you're using Dynamic Dialog). Normally, you will see OK and Cancel buttons along with slots for the sample number and MIDI channel. The MIDI channel is normally set to the sampler's base channel. Other controls may be included to take advantage of certain features which a particular sampler may offer, such as wave banks or sample name catalog facilities (for details on individual samplers, see the Samplers section). Set the slots and buttons to reflect the wave you'd like to receive from the sampler. To start the transfer, select the OK button. If the sampler can send the requested wave and the computer can accept it (ie, has enough memory), the transfer will start.

Once the data transfer begins, the mouse pointer will turn into a "Receiving" sign. If at anytime you would like to halt the data transfer, just click on the small Abort window. This might happen if you suddenly realize that you're transferring the wrong wave, or that you forgot to turn the sampler on! Transferring large amounts of MIDI data is a rather busy job. We don't recommend trying to run other programs at the same time as there may be a noticeable slow down. If any errors are caught during transmission, you will generally be notified of this (some samplers handle this pretty well on their own, some don't). Once the data transfer is complete, the mouse pointer will go back to its default shape, and the new wave will be drawn in the editor's window. You may now do with this wave as you wish.

Sending a Sample Dump

The sending process is very similar to receiving a sample dump. Click on the **S** button in the editor's toolbar, or select **File/Send**. As with the Receive function, a sampler-specific MIDI window or dialog will open allowing you to set the desired parameters for the given wave, such as sample number and MIDI channel. Once again, the MIDI channel is usually set to equal the sampler's base channel. Some samplers may require that the target sample number be cleared (erased) on the sampler before it accepts the new wave. Select the OK button to start the data transfer.

If the sampler can accept the wave, the mouse pointer will turn into a "Sending" sign. If you need to abort the sample dump, just click on the small Abort window. Once the send process is complete, the mouse pointer will revert to its default shape.

Standard 12 (SDS) driver

This is the 12 bit version of the MMA Sample Dump Standard protocol. It supports only sustain loops, ignoring release loops. Several companies make samplers which conform to the MMA guideline. We have not had occasion to test all of them. The list includes:

Dynacord AD-1, AD-2, ADS, ADS-K.
E-mu Emax, SP-1200.
Korg M1 and M3 (with Frontal Lobe), T1, T2 and T3 (with PCM card).
Oberheim DPX-1, Prommer.
Sequential Circuits Prophet 2000/2002, VS, Studio 440

Simmons SDX.
Yamaha TX 16W.

If you have difficulty receiving sound samples from your sampler, try checking the **Tell Sampler to Wait Between Packets** checkbox. This tends to make the transfer more reliable but also slows down the process.

The Prophet 2000 is a very quirky keyboard as far as its MIDI implementation is concerned. It does strange things to a wave whose size is not based upon a particular factor, including perhaps chopping off the end of the wave and maybe adding audible garbage (clicks, and parts of overwritten data). A special driver is available for the Prophet 2000/2002 which compensates for the prophet's weirdness. (See Prophet 2000 driver).

The Yamaha TX 16W needs to be in an appropriate receive mode before accepting data. Check that the Yamaha's device number setting matches up with the MIDI Channel in the dump dialog. Data following the end of the Sustain loop is ignored by the Yamaha itself. (You may or may not want to move the Sustain loop's end to the very end of the wave before transferring.) The initial OS shipped with the first few TX 16W units had a broken MIDI implementation. The wave that it sent to the computer had virtually every sample point clipped. The second OS fixed this.

While the E-Mu E-Max series does use the MMA standard sample dump protocol, it is important to remember that it 'numbers' it's samples not in the order stored (as most samplers do), but by the MIDI note number for the root pitch it is mapped to (primary voice). For example, if you have a sound mapped on the E-Max's keyboard such that its root pitch is note number 60, you must request this as sample number 60 from Wrench's sample transfer dialog.

Standard 16 (SDS) driver

This is the 16 bit version of the MMA Sample Dump Standard protocol. It supports only sustain loops, ignoring release loops. Most recent vintage samplers use this protocol, so if you're not sure about your sampler, this is the place to start. The list includes:

Akai S-1000 series and above.
AKG ADR 68K.
E-mu EmaxII, Emulator III, and above.
Forat F-16.
Peavey DPM-SX, DPM-SP series.
Roland S-750, S-770, and above.
Sequential Circuits Prophet 3000

The Roland S-750/770 requires that your sample have a sustain loop for it to accept the sample. If you load a sample from some other source and send it to the Roland, you will get an error if this sample does not have a sustain loop. You can verify whether or not you have a loop by simply looking at the Info requester. If the "Sustain" slot is empty, you don't have one. You can make a sustain loop by using the menu item Loops+Markers/ Loop Set.

Akai S612 driver

This driver is for the Akai S612 sampler. The S612 was one of the first MIDI samplers on the market. By today's standards, it is somewhat limited in power. This driver works with version 1.1 of the Akai software. The S612 does not allow for the transfer of large waves or (ouch) the sampling frequency. The sample rate is assumed to be 16 KHz (the S612 default), although you may easily change it using the Fs Transposition function within Wrench.

Since the S612 can hold only one sample at a time, there is no wave number slot on the dump dialog. The S612 imposes an upper limit of 9 on the MIDI channel.

When receiving a dump from the S612, a wave of approximately 32,000 points will be created. Looping points are saved. When sending to the S612, if the source wave is greater than 32,000 points, only the first 32,000 points will be sent. Looping points cannot be sent back to the S612. (Even if they could be sent it would do little good, since the S612 requires that its playback breakpoints be on multiples of 256 sample points.)

A complete sample dump takes about 30 seconds in either direction. When sending to the S612, the S612's LED display will flash. When receiving from the S612, its display will brighten.

Korg DSS1/DSM1 drivers

These drivers is for the Korg DSS-1 and DSM-1. They are very similar so we'll just refer to the DSS-1 here. The dump dialog has a Sound number slot and a MultiSound number slot. Using Korg nomenclature, the MultiSound number corresponds to the desired DSS-1 multisound, while the Sound number corresponds to a sound within the multisound.

To send a sound to Sample Wrench, set the desired Sound, MultiSound, and Channel numbers, and then select OK. In a moment, the mouse pointer will turn into Receiving, which indicates that information is being transferred.

The process of sending a sound to the DSS-1 is similar, except that the MultiSound and Sound numbers will be ignored (with one exception, detailed below). Once MIDI contact is made, the DSS-1 front panel will read "Load Completed". Once all of the sound data has been sent, the DSS-1 will read "Write Requested". The single sound will be set up as a multisound on the DSS-1 and the present program will be altered to use this new multisound. The DSS-1 OrigKey will be set to Wrench's RootNote, and the TopKey will be set Wrench's HighNote.

When you send a sound over to the DSS1, it is possible to overwrite an existing sound (leaving all of the other sounds intact), instead of creating a brand new multisound. This will only work if the sound you are sending is no larger than the sound you are trying to replace. If you are not sure (and don't mind mind possibly losing the tail end of the sound) you can select the Chop checkbox before sending. This will force the sound to fit into the DSS1. If the sound is larger than the target and you don't select Chop, then a new multisound will be created, with this sound as sound number one. The overwrite capability is very useful if all you plan on doing is using Wrench to alter loop points, scale, EQ, or perform similar functions which don't alter the length of the sound. In this way you can pull out individual sounds, alter them, and then send them right back to the DSS1 without having to remap everything.

A full memory dump of the DSS-1 takes a little over 2 1/2 minutes to transfer. Smaller sounds take considerably less time. (The maximum transfer rate is a function of the MIDI specification, and not Sample Wrench or the DSS-1).

Ensoniq EPS/ASR drivers

These drivers are for the Ensoniq EPS and EPS16, and the ASR-10 and ASR-X. For the most, part, these drivers look and act the same. On the dump dialog are text slots for the EPS/ASR WaveSample number (ie, Wave Number), Layer number, and Instrument number, and one for the MIDI Channel number.

The EPS must have its system exclusive ability set to "on". This is set by hitting the Edit and MIDI keys on the EPS front panel, and then scrolling until MIDI SYS-EX is found. The very first time that a send or receive is performed, the EPS will need to load an "overlay" from floppy disk. The front panel will light up when this happens, and instruct you to insert an operating system (OS) disk and then hit the enter key. After you do this, the EPS will inform you that the operation was successful by displaying "Overlay Loaded". The sample dump will continue on normally at this point. You can make your life a little easier if you leave a disk containing the OS in the disk drive. You can tell when a sample is being transferred, because the EPS display will flash "MIDI" and the main status line will flash about once every second.

Normally, the sequence of events for editing requires that you send the sound from the EPS to the computer, edit it, and then send the sound back to the EPS. When Wrench sends a sound to the EPS, it uses the target sample's parameters as a basis, only modifying such things as sample size, loop points, sampling frequency, and the key range. In other words, your initial settings for filters, volume, and such are left unharmed. Given this fact, you might wonder if it's possible to create brand new Instruments, Layers, and WaveSamples from scratch. The answer, of course, is yes! Simply check the box marked "Create" next to the appropriate Instrument, Layer, or Wavesample in order to create it. Please note that the EPS does not let you arbitrarily create Layers and WaveSamples in any order, they must be produced sequentially. In other words, if you are creating a new Instrument, the Layer and WaveSample values will be 1. (Wrench will do this for you if you forget.) Also, as another convenience, only the highest order element needs to be set (checked). Wrench is smart enough to realize that creating an Instrument implies creating a new Layer and WaveSample as well. Here are some examples where * means "Create box checked":

Instrument	**Layer**	**WaveSample**	**Meaning**
*2	*1	*1	Create Instrument 2, Layer 1, WaveSample 1. Instrument 2 must not already exist.
*2	4	2	Same as above since Wrench will override the Layer and WaveSample values (this is a new Instrument.)
3	*1	*1	Create Layer 1, WaveSample 1 in Instrument 3. Instrument 3 must exist, Layer 1 must not already exist.
3	*1	5	Same as above due to Layer override of WaveSample (this is a new Layer.)
3	*4	*1	Create Layer 4, WaveSample 1 in Instrument 3. Instrument 3 must exist, Layer 4 must not already exist, but Layer 3 must exist since sequential ordering is required.
5	2	*1	Create WaveSample 1 in Instrument 5, Layer 2. Instrument 5 and Layer 2 must exist, WaveSample 1 must not already exist.
5	2	*4	Create WaveSample 4 in Instrument 5, Layer 2. Instrument 5 and Layer 2 must exist, WaveSample 4 must not already exist, but WaveSample 3 must exist since sequential ordering is required.

Don't worry if you screw up the numbers. You will just get a message back saying "Illegal Layer Number" or something similar. You won't lock up the computer or the EPS. Just try it again.

While MIDI data is being transferred, the EPS front panel will flash "MIDI". If this flashing stops for several seconds (over 15 or so) and the mouse pointer is still showing Sending or Receiving, an error has probably occurred. To abort this condition hold down the left mouse button and then release it.

After a transfer, the EPS will automatically be set to edit mode for you so that you can then define and "tweak" parameters as needed.

Legal Data Ranges

Instruments and Layers: 1 through 8
WaveSamples: 1 through 127
MIDI Channel: 1 through 16
Name: only the first 12 characters will be used

If you enter a crazy number (like 4023 for Channel number), Wrench will limit the number to the legal maximum.

The EPS can be somewhat cranky. We suggest that once the sampler is set up and ready for transfer that you insert the OS disk into the EPS's drive (since it will need it to load an overlay) and that you put the EPS in edit mode. You can then initiate a sample send or receive. Remember, the EPS's display will flash while receiving/sending MIDI info. If the EPS stops flashing for more than a minute or so and Wrench still shows the Sending or Receiving pointer, the EPS has probably gotten lost or overloaded and you should terminate the transfer from Wrench. In such a case, the EPS will have only sent or received a portion of the wave. Try to send or receive the wave again. If this does not work, try reloading the sample into the EPS from its disk, and transferring via Wrench again. If this still does not work, or if the EPS locks up, turn the EPS off and start over.

Sequential Prophet 2000 driver

This driver is for the Sequential Circuits Prophet 2000/2002 with version 3.0 software (or later). The prophet dump dialog looks very much like the standard 12 driver. The Sound Number is the desired Prophet Sound Number (1 to 16). Be aware that, unless the Prophet is in OMNI MIDI Mode (0), each Prophet preset has its own MIDI channel setting, and changing a preset can change the Prophet's channel.

The Prophet display will show an "r" as Wrench is sending it a wave, or a "d" as the Prophet is sending a wave to the computer.

If you're transferring a wave from the computer to the Prophet, then after you select OK, another dialog will pop up and tell you to set the Size via the Prophet's front panel. If you don't set the Sound Number's Size on the Prophet, the Prophet may chop off the end of the wave and terminate (without telling Wrench). For example, you may see a dialog appear that says:

Set Prophet Size = 18

At this point, you should go to the Prophet's front panel, select the Size parameter (next to Sound Number in the upper left), set the value to 18, and press Execute. (Do not change the sound number!) You can then answer OK to Wrench's dialog, and the transfer will continue. Note that the maximum wave size that an expanded Prophet can accept is 512 blocks (524,288 sample points).

One weird thing about the Prophet is that it insists that the size of an incoming wave be based upon a certain factor. On a typical wave, this factor may be off by 1 to 58 sample points. What the prophet may do is refuse to accept these last points, thus chopping off your wave by a slight amount and perhaps ruining a loop. Upon sending, the prophet driver will pad out your wave to this factor in order to compensate for the Prophet's weirdness. Thus, your wave will not be chopped. Indeed, if you then ask the Prophet to send the wave back, it will be slightly larger than the original. Unfortunately, the Prophet also exhibits the same weirdness when it sends a wave to the computer. It may not send as much as the last 58 points of a wave. You don't have to worry if you're receiving a wave that Wrench sent to the Prophet since Wrench has already seen to it that the wave's size is appropriate. It is important though, that you never alter the Prophet Start or End parameters and then do a Recover Memory on this wave as this will cause the Prophet to alter the size and perhaps instigate the weirdness. Of course, if you're receiving sounds that were made by someone else who did use the Prophet's parameters to size the sample, then you may end up with a little lost data at the end. If you ever sample on the Prophet, don't adjust the Start/End/Recover Memory via the Prophet. Instead, send the raw sample to Wrench, chop it as desired, and send it back to the Prophet. Wrench will fix the size just fine.

The Prophet allows only 3 sampling Rates; 15625, 31250, and 41667 Hz. If your wave's Sample Rate is not one of these, the driver will set the Prophet's Rate to the next highest rate (or 41667 if the rate is higher than this). That will mean that the wave will play back at a different pitch than it would at its original rate. You can either choose to adjust your mapping, or use Wrench's Sample Rate Transpose function to set to one of the Prophet's 3 rates before transfer.

SMDI (SCSI Musical Data Interchange) driver

This driver is for samplers which adhere to the SMDI specification for high speed SCSI transfer of sound data. This includes the Peavey DPM-SX/SP series, E-Mu ESI-32 and E64K, and Kurzweil K2000 series. Not all SCSI equipped samplers are SMDI compliant, so check with the sampler manufacturer to be sure. The main advantage of SMDI is speed. It is about 50 to 100 times faster than a MIDI transfer.

This driver requires an ASPI compliant SCSI controller in your computer, such as those made by Adaptec (and others). Follow the cabling and external device hook-up instructions which came with your controller, being particularly careful to avoid conflicting SCSI IDs. Connect your SMDI device to the controller, turn the SMDI device on, and then boot the computer. On this initial boot-up, Windows should detect the new SCSI (SMDI) device and prompt you for device driver installation. Select "Ignore (Windows will not prompt you again)". Depending on the system, you may have to select this several times in a row. (Note that your SMDI device will be listed under "Other Devices" in the Control Panel’s System Device Manager.)

Using the Send/Receive dialogs is straight forward. There are text slots for Adapter number (if you have more than one controller installed), SCSI ID, LUN, and sample number. Wrench will auto-detect available SMDI devices and place these in a drop-down list. Selecting a listed item will automatically transfer the proper settings to the text slots for you. From there it is just a matter of selecting OK to start the transfer.

Frequently Asked Questions About Wrench

Q: I don't have any MIDI samplers, can I use Sample Wrench with just my audio card?

A: Wrench does not require a MIDI sampler and many people use it as a sound card audio editor. Almost any audio card will work, assuming it conforms to Windows multimedia standards. Not all sound cards are created equal though. For example, a given card may only be able to play or record at certain sample rates. Wrench tries to work around some limitations, but it requires that the sound card be capable of recording and playing at 16 bit resolution. (Some early or less expensive models can only achieve 8 bit resolution).

Q: Can I use Sample Wrench to edit sounds for my multimedia project?

A: Most certainly. Most of the multimedia authoring programs in the WIndows market deal with sounds in the .WAV file format. Wrench can save and load these files. Very often, these are of the 8 bit variety. The advantages of using Wrench over a simpler 8 bit audio editor are many. First, by editing with 16 bit or higher precision, truncation and rounding noise is minimized. Second, you have access to many powerful DSP functions including the advanced equalization section, both amplitude and time compression, pitch shifting, accurate sampling frequency transposition, and much more. Third, Sample Wrench gives you a gateway into the large collection of sounds available for other platforms. Finally, Wrench allows you to transfer sounds to and from various MIDI based sample modules and keyboards which offer 16 bit fidelity.

Q: Can I use Sample Wrench with my sequencing program or hard disk recorder?

A: Yes. Many programs of this type can access sound files using the .WAV style. Generally though, they do not have many of the editing features or sampler communication abilities of Wrench. So, you can use Wrench to edit or import sounds, and then export them to the sequencer/hard disk recorder. To do so, simply make sure that the File Format is set to WAVE style. For professional use you'll want 16 bit resolution, and for less demanding work you might opt for 8 bit resolution in order to keep the files smaller. Be aware that some audio programs can only work with sound files using the standard rates of 44.1 kHz, 22.05 kHz, or 11.025 kHz. If the sound was not recorded using one of these rates, you may have to use Wrench's Sample Rate Transpose function in order to reset it.

Q: Can I import sounds from other, older computer platforms?

A: Yes. Sample Wrench supports AIFF, 8SVX and Sound Designer Type 1 file formats which were popular on other computers. The 8SVX and Sound Designer formats are limited to mono sounds. (For stereo sounds, you can load two mono sounds into Wrench and then combine them into a single stereo sound.) Wrench also supports several types of RAW files. These allow you to import just about any mono linear PCM sound file with just a little work on your part.

Q: Is there a way to customize the toolbar on the editors?

A: Yes. Simply double-click on the empty area right next to the last button. The system standard toolbar editor dialog will pop up. You can add, delete, or reorder the buttons using basic drag and drop techniques. This dialog also includes context sensitive help.

Q: Can I have Sample Wrench automatically use my preferred operating environment (editor colors, axis units, file format, etc.) when it first starts?

A: Yes. There are two approaches to this. First, Wrench uses the Windows Registry to keep track of your last settings. This is great if you want the program in the same state as when they left it. On the other hand, you may want Wrench to open the same exact way each time, no matter how you last left it. In this case, you'll want to use config files. When Wrench is first launched, it looks for two files in the present directory. They are wrench.config and wrench.macro. wrench.config holds information on your color scheme, the editor attributes, your default sampler, and more. wrench.macro holds the names of the Enable macros which have been assigned to your function keys. Once you have your operating environment set the way you like, select Save Config from the Setup menu. Save this file as wrench.config in the same directory as Wrench.exe (i.e., the Wrench Program directory). Once you have your macros set up, save them as wrench.macro using the Save Macros menu item. Now, when you start Wrench, these files will be loaded automatically and you'll have the operating environment of your choice.

More advanced users can place different versions of these files (with the same names) in different directories. Depending on which directory Wrench is launched from, you can have different automatic setups. You can aso load these configs whenever you need a specific environment.

Q: Can I get Wrench to automatically start a bunch of things for me each time it starts up?

A: Yes, this is done viable the Enable scripting language. On startup, Wrench looks for a special Enable script called wrench.macroinit. If Wrench finds it, the script is executed. This script can access just about any of the 80+ Wrench specific functions or well over 100 Visual Basic compatible functions/statements.

Q: I need to translate a whole bunch of sound files from one format to another. Is there a way I can automate this?

A: You bet. Things like this are easy to accomplish using Wrench's Enable scripting language. In fact, you'll find an example of how to do a batch translation (and a lot more) on disk.

Q: Can I trade samples between different samplers?

A: Yes. Sample Wrench can be thought of as communication hub in this sense. When you buy Sample Wrench, you get support for a wide variety of samplers. To communicate with any given sampler, all you need to do is specify the proper driver. This is done through the Sampler menu item. For example, say you want to transfer a sound from an Ensoniq EPS over to a Peavey SP. After you have hooked in the EPS and have Wrench up and running, select Sampler. From the menu listing select Ensoniq EPS. To receive the sample, open an editor and select the R button in the toolbar of the editor window. A dialog will open in the editor allowing you to specify which sample you want. Once this is set, select OK. The transfer will start. Once it is done, the sample will be drawn in the editor window. You can now edit this sample, save it, or do as you like. When you are ready to transfer the sound to the SP, swap the MIDI cables over to the SP. Now, select Sampler. From the menu listing, select Standard 16 Bit. The SP is a 16 bit device which conforms to the MMA standard sample dump protocol so there is no need for a special SP driver. (If you have questions about any given device, see the Sampler Communication section of the manual). To send the sample to the SP, select the S button in the toolbar of the editor window. A dialog will pop up allowing to set where the sound will go. Once this is done, select OK and the transfer will start. Once the transfer is completed, you can play the sound from the SP, save it to an SP disk, or whatever you wish.

Q: Is there a way where I can just sweep my mouse pointer over the waveform in order to define the area I'd like to edit?

A: Yes, there is- in fact, there are two ways of doing this. The first way is to simply choose Affect Mouse from the Setup menu. The edit Affect area is then defined by placing the mouse near the area of interest, pressing the left mouse button and sweeping the mouse over the desired area. The process is completed by releasing the mouse button. The Affect area is drawn in reverse highlight (and if Overviews are active, the Affect area will be shown by a thin highlighted bar on the top edge of the Overview).

The second technique is a little different, and some people prefer it since the edit Affect area doesn't wind up being highlighted (to some people this is distracting). Using this technique, you'll be able to define any edit area very quickly, see a close-up of it, and still know where you are with respect to the waveform as a whole. To do this, set the Affect menu item to "In View" (under Setup). Also, make sure that you have the "Overviews" menu item active (under View). What this means is that the edit window will be split into two chunks: an upper portion which shows the entire waveform, and a lower portion which just shows the area you've zoomed into. It is this lower portion which will be edited. Remember, when you zoom into a wave, the corresponding portion of the overview will be highlighted so you know exactly where you are at all times. To define the zoomed-in area quickly, use the Zoom Box in the overview. Yes, that's right- the Zoom Box will work in the overview as well as in the main portion! Once you have the above settings in place, defining the edit area is as simple as moving the Zoom Box over the area of interest in the overview. Once the mouse button is released, the lower portion is redrawn to the area you've defined, ready for an edit. The nice thing about this technique is that you get a zoom-in of the edit area and can still see the entire wave. Also, to define a new area to edit, you simply move the Zoom Box over the new area inside of the overview. This new area can be completely different from the first area defined.

To take this technique one step further, if you know that there are a handful of areas which need editing and which you'll need to bounce back and forth between, you can use the Set View and Get View menu functions to remember them for later recall in the session. Finally, please take note that you can stay in Zoom Box mode and still use almost all of Wrench's features- you don't have to go back to Normal mode. (The only things you can't do in Zoom Box mode are use the X-Y Readout or the ability to grab markers and loops since they require Normal mode, use the Freehand pencil since that requires Freehand Draw mode, or use Scrub since that requires Scrub mode).

Some Other Weird Things You Can Do

With just a little imagination, you can get Wrench to perform some interesting alterations on your sounds which you might not have thought possible. The trick is to use different functions in unique combinations. Here are a few ideas:

Backwards echo and/or reverb are both interesting and easy to create. Instead of having the echoes appear after the sound, you can get them to appear before it. In a similar manner, you can get reverb to swell up into a sound, rather than having the reverb trail after it. The trick is to make sure that you have some silence before the area of interest, and before applying echo or reverb, reverse the sound. You'll then have a backward sound with normal trailing echo or reverb. By reversing the sound again, the sound comes out normally, but the echoes or reverb appear before the sound.

Speaking of reverb, if you subtract the original sound from a reverberated version, you wind up with just reverb. Subtracting can be created by either adding the original after inverting it, or you can use L-R stereo to mono conversion where the stereo sound is simply dual mono with only one channel previously affected by reverb.

For a strange, almost vocoder-like effect on human voice, use Impulse Modeling with a non-traditional impulse such as a single piano or guitar note.

For a neat positional effect, try turning a simple mono sound into stereo (ie, dual mono) and then inverting one channel. This has a decidedly different quality through headphones than ordinary dual mono. Once this is done, you can alter the effect further by adding some reverb.

In order to make a dual mono sample sound a little more like natural stereo, try adding reverb to each channel individually, using slightly different loudness, diffusion and pre-delay settings. For something unnatural, try adding reverb or echo to just one channel. In a similar vein, try using different chorus and flange settings for each channel.

And Now For Something Completely Useful...

Very often, folks make commercials which consist of a narration/ voice-over which resides on a bed of background music or sound effects. The narration may be a mono track centered between the two stereo channels. If this main track is a bit on the long side, it needs to be time compressed to fit the length of the commercial. Wrench's Time Stretch function is designed with exactly this in mind. If the narration and background music do not have to be in sync, which is often the case, you can optimize the results by treating the two parts as separate entities. Since the background music is somewhat flexible in time (ie, you use whatever amount you need), there is no requirement to time compress it. You will generally get better results if you time compress the narration by itself, and then add it to the background music, rather than adding the two and then compressing the entire mix. The simpler the waveforms involved, the more accurately Wrench can render the result, and a narration is much simpler than narration plus music.

Sample Wrench V5 Enable Scripting

Introduction and Sound Editing Specific Functions Reference

Enable is the script/macro language used by Sample Wrench. It allows you to automate complex or repetitive tasks (batch processing), create alternate ways of interacting with Wrench, or produce your own personal Sample Wrench "plug-ins". Enable is Visual Basic compatible. If you know how to use Visual Basic, then you know how to use Enable. If you are not familiar with Visual Basic, please refer to the generic Enable documentation which is available at http://www.dissidents.com/download/WrEnable.zip. If you prefer, books and guides which explain Visual Basic should be available at your favorite bookstore. Also, **you can have Sample Wrench generate macros for you, automatically** (see below). The remainder of this section covers the Wrench-specific portion of Enable and assumes that you have at least some familiarity with Basic or scripting languages in general.

Enable macros are in essence, Basic programs. You can think of Enable as a version of Visual Basic which contains about 100 new functions for accessing and editing sound samples. The macros are created using your favorite text editor (such as Notepad). A set of macros is assigned to function keys F2 through F12 using the Assign Macros item under the Setup menu. Once assigned, the macros are started by pressing the desired function key. Sets of macro filename assignments can be saved for future use and reloaded into Wrench via the Save Macros and Load Macros items found under the Setup menu. Wrench also automatically loads two files for you when it starts up (assuming they're available). The first one Wrench looks for is wrench.macro. This is a set of filename assignments as created by Save Macros. To create your own wrench.macro file, fill in the key assignments using Assign Macros and then save it using Save Macros. Specify wrench.macro as the save filename.

Besides the keyboard assignments, Wrench also looks for a file called wrench.macroinit. This is an Enable script which you write. Wrench will run this script for you automatically. This script can do many things such as open editors, load waves, alter editor attributes and so on. When used in conjunction with the auto-load configuration file (wrench.config), you can have Wrench open up exactly the way you want it. (For advanced users, note that it is possible to load configuration and macro files from an Enable script. Thus, you could have your wrench.macroinit script ask you which configurations or assignments you wanted if you regularly use different ones.) Sample Enable scripts and macro files are included on disk and later in this document.

Automatically Generating Macros

Many times you may like to repeat a number of steps on different files and you don't want to have to resort to writing a macro from scratch. For these cases, Wrench can generate a macro for you by recording your sequence of actions. The resulting script can then be assigned to a function key and used as is, or you can use it as a skeleton or starting point for a more complex macro. This is done through the **Setup/Macros/Auto-Record Macro** menu item. Selecting this will open the standard file save dialog so you give the macro an appropriate filename (normally ending in ".bas"). At this point, Wrench will start recording your actions and writing directions to this file. When the set of actions is complete, simply re-select the menu item (note that it will now say **Stop Recording**).

For example, let's say that you wish to create a macro which applies a 3 dB treble boost above 5 kHz, reverses the sound, and then maximizes the level. Here's what you do: once you have loaded the first sound to work on and selected the Affect area, select Setup/Record Macro. Give the macro a decent name like "treblereverse.bas". For ease of use you may wish to select one of the Auto-assign buttons (let's say F2). This assigns the macro to the function key automatically so you don't have to call up the Assign Macros dialog. Now go through the editing process. First, call up EQ: Treble and set it for Use Treble, 5000 Hz, and 3 dB boost. Select OK. Your sound has been EQ'd. Now select Reverse, and then Maximize. You have now completed the editing so select Setup/Stop Recording. This three step macro has been written for you and assigned to the F2 key. To perform this same set of actions on another part of this file, simply select the desired Affect area and then hit F2. That's all there is to it. If you wish to use this macro on another file, load that file into the editor used to create the macro, select the Affect area, and hit F2. If desired, you can use a text editor to alter the macro file ("treblereverse.bas"). One common alteration is to change the editor number used for the function calls to 0. An editor of 0 means "use the currently active editor". The recorder always uses the actual editor number. This is so that you can create macros which use complex interactions between several editors. It is also useful if you simply want to create a journal file (i.e., a transcript of all the processing done on a particular sound file). Journals can be invaluable if you're designing/recreating wacky and far-out special effects.

About Enable Functions

The functions listed below are presented in two major groups: Items from the Functions and Effects menus, and items which are of a more general nature. Within each group the functions are listed alphabetically. All Wrench-specific functions use an 'sw' prefix to distinguish them from ordinary Basic functions.

Each function is presented with its template. The template includes the name of the function, its argument list (ie, the items it needs to perform its job), and the type of value it returns to you. Almost all functions use long integer or double floating point variables and return a long value indicating success (1) or failure (0). & is used to denote a long integer argument (a whole number), # to denote a double precision 64 bit floating point argument (a value which may contain a fractional part), ! to denote a single precision 32 bit floating point argument, % to denote a short integer argument (16 bit) and $ to denote a text string argument (some mix of letters, numerals, and so forth). In general, most functions require the ID number for the wave which you wish to act on. This item is noted in the argument list as edid. For example, the Gain function appears as follows:

swGain(edid&, db#) As Long

This tells you that the name of the function is swGain, the first argument is an integer indicating the editor number, and the second argument is a floating point value which sets the gain in decibels. The function returns an integer to indicate success/failure. In order to produce a gain of 3.2 dB in editor number one, use:

r = swGain(1, 3.2)

if r comes back as 1, then the process was a success. A function may be unsuccessful if the specified editor is not open, the editor is empty (no wave loaded), or there isn't enough memory.

Editor IDs range from 1 through 99. This is the number you see in an editor's title bar. If you open an editor via Enable, this value is returned to you (see swOpenEditor). You can also use 0 for the ID. This is shorthand for saying "use whatever editor is presently active" (ie, the one whose titlebar is highlighted). This is known as the default Enable editor.

A few words for advanced users

A script may be halted if some sort of error message needs to be displayed. For a batch process this may not be desired. You can suppress these message boxes by redirecting error messages either to the bottom status bar or to a log file (see swSetMsgType, swOpenMsgFile, swCloseMsgFile). You can also add your own messages to the log file (see swWriteMsg). Finally, note that all items are passed by value (ie, ByVal).

Sample Wrench V5 Enable Scripting

General Purpose Functions

For details on the following, see the main Sample Wrench documentation. The variable 'r' in the examples is the return value which can be checked for success/fail, prior values, etc. The examples presented are not complete scripts, just simple segments of scripts. Complete examples are given at the very end of this section and on disk.

swBackupEditor(edid&) As Long

Forces a backup of the wave data for the specified editor. Useful for "build your own" DSP functions: call prior to editing. Note that if the Backups menu item has not been selected, this call will do nothing.
Returns success/fail.
See also: swCleanUpEditor

swBackups (depth&) As Long

Sets the maximum number of levels in the Undo History list. 0 means no backups, no Undo.
Returns success/fail.

swCleanUpEditor(edid&) As Long

Resets mouse pointer, initializes status bar and progress bar, limits loops and markers, redraws the wave, and adjusts certain internal variables. Useful for "build your own" DSP functions: call after all processing is complete.
Returns success/fail.
See also: swBackupEditor, swSetProgressBar, swResetPointer, swSetStatusMsg, swSetWaitPointer

swCloseEditor(edid&) As Long

Closes the specified editor. edid is the editor number to close.
Returns success/fail.
Example: Shut down editor number one.

```
        r = swCloseEditor( 1 )
```

See also: OpenEditor

swCloseFindFile() As Long

Used to finish a filename search started by swFindFirstFile. This function must be called when you have completed a search using swFindFirstFile in order for Sample Wrench to perform internal housekeeping chores.
Returns success/fail.
See also: swFindFirstFile, swFindNextFile

swCloseMsgFile() As Long

Closes the log file which was opened via swOpenMsgFile.
Returns success/fail.
See also: swOpenMsgFile, swWriteMsgFile

swDeleteWave(edid&) As Long

Deletes the wave in the specified editor. If the wave has unsaved changes, a message box will pop up allowing the user to save, ignore, or abort.
Returns success/fail.
Example: Remove the wave from the default editor (this does not delete it from disk).

```
        r = swDeleteWave( 0 )
```

See also: swDeleteWaveNV

swDeleteWaveNV(edid&) As Long

Deletes the wave in the specified editor without asking for verification.
Returns success/fail.
See also: swDeleteWave

swFindFirstFile(searchstring$) As String

Used to start a scan of a directory for filenames. searchstring is a directory path and filename which may include the * and ? wildcards.
Returns the first filename that matches searchstring. If there are no matches, an empty string is returned. This function will also return the names of matching subdirectories, although it will not search the contents of the subdirectories.
Example: Start a search in the directory C:\MySounds for any file with the .voc extension.

```
        f = swFindFirstFile("C:\MySounds\*.voc")
```

See also: swFindNextFile, swCloseFindFile

swFindNextFile() As String

Used to continue a scan of a directory for filenames as started by swFindFirstFile. Typically, swFindNextFile is placed in a while loop, continuing to extract filenames until the function returns an empty string.
Returns the next filename that matches searchstring. If there are no matches, an empty string is returned. This function will also return the names of matching subdirectories, although it will not search the contents of the subdirectories.
Example: Display in a pop-up window all of the files in the directory that match swFindFirstFile's searchstring.

```
        f = swFindFirstFile("C:\MySounds\*.voc")
        while f <> ""
                MsgBox "Filename: " & f
                f = swFindNextFile()
        wend
        swCloseFindFile()
```

See also: swFindFirstFile, swCloseFindFile

swGetActiveEditorID() As Long

Returns the presently active editor number, or 0 if none are available.

Example: Find the active editor and then close it.

```
        id = swGetActiveEditor()
        r = swCloseEditor( id )
```

See also: swGetNextOpenEditor

swGetNextOpenEditor(x&) As Long

Returns the next editor which is open whose ID is at least as large as x (or 0 if none are available). This is useful when looping through an unknown number of editors.
See also: swGetActiveEditorID

swGetSampleValue(edid&, offset&) As Integer
swGetSampleValueR(edid&, offset&) As Integer

Retrieves the value of the sound waveform at the the specified offset from start. offset ranges from 0 through the wave's size - 1. swGetSampleValueR is similar, but retrieves the value from the right channel (if one exists). The return value is a 16 bit integer which ranges from -32768 up to +32767.
Returns value of waveform.
See also: swInquire, swSetSampleValue

swGetSampleValueFloat(edid&, offset&) As Single
swGetSampleValueRFloat(edid&, offset&) As Single

Retrieves the value of the sound waveform at the the specified offset from start. offset ranges from 0 through the wave's size - 1. swGetSampleValueRFloat is similar, but retrieves the value from the right channel (if one exists). The return value is a 32 bit single precision float which ranges from -1.0 to +1.0.
Returns value of waveform.
See also: swInquire, swSetSampleValueFloat

swGetVersionInfo() As String

Returns version and time text string.
Example: Display the version info in a message box.

```
        s = swGetVersionInfo()
        MsgBox "The version is: " & s
```

swInquire(edid&, item&, lmid&) As Long

Determines certain characteristics about the wave loaded in the editor.
item may be any of the following:

SW_INQUIRE_CHANNELS	0 if empty, 1 if mono, 2 if stereo (lmid arg is ignored)
SW_INQUIRE_LOOPEND	position in samples, for the loop specified by lmid
SW_INQUIRE_LOOPSTART	position in samples, for the loop specified by lmid
SW_INQUIRE_MARKER	position in samples, for the marker specified by lmid
SW_INQUIRE_PERIOD	in nanoseconds (lmid arg is ignored)
SW_INQUIRE_RATE	in Hertz (lmid arg is ignored)
SW_INQUIRE_RELEASE	Release loop ID (lmid arg is ignored)
SW_INQUIRE_SIZE	in samples (lmid arg is ignored)
SW_INQUIRE_SUSTAIN	Sustain loop ID (lmid arg is ignored)
SW_INQUIRE_AFFECTSTART	offset of edit Affect start, from 0 (lmid arg is ignored)
SW_INQUIRE_AFFECTEND	offset of edit Affect end, from 0 (lmid arg is ignored)

SW_INQUIRE_AFFECTLEFT	1 if left channel edit, else -1 (lmid arg is ignored)
SW_INQUIRE_AFFECTRIGHT	1 if right channel edit, else -1 (lmid arg is ignored)
SW_INQUIRE_FORMAT	code number for original format (SW_FORMAT_...)

Returns the desired info, or -1 if not available.
Example: Find out how large the wave in editor 2 is.

```
        size = swInquire( 2, SW_INQUIRE_SIZE, 0 )
```

See also: swGetActiveEditorID, swIsEditorLoaded, swIsEditorOpen, swSetFormatType

swIsEditorLoaded(edid&) As Long

Indicates whether an editor contains a wave (1), or is empty (0).
Returns success/fail.
See also: swGetActiveEditorID, swInquire, swIsEditorOpen

swIsEditorOpen(edid&) As Long

Indicates whether or not an editor open.
Returns success/fail.
See also: swGetActiveEditorID, swInquire, swIsEditorLoaded

swKeyMap(edid&, low&, root&, high&) As Long

Sets the key range of a sample. low, root, and high are in MIDI note numbers, from 0 through 127.
Returns success/fail.

swLoadConfig(path$) As Long

Loads the configuration file specified by path.
Returns success/fail.

swLoadEnvGenPreset(path$) As Long

Loads the Envelope Generator curve file specified by path.
Returns success/fail.
See also: swEnvelopeGenerator

swLoadMacros(path$) As Long

Loads the macro file specified by path.
Returns success/fail.
Example: Load the macro file c:\wrench\mymacros.mcr

```
        r = swLoadMacros( "c:\wrench\mymacros.mcr" )
```

swLoadPlayPrefs(path$) As Long

Loads the playback preferences file specified by path.
Returns success/fail.

swLoadTransFuncPreset(path$) As Long

Loads the Transfer Function curve file specified by path.
Returns success/fail.
See also: swTransferFunction

swOpenEditor() As Long

Opens a new editor window.
Returns the editor number ID, or 0 on failure.
See also: swCloseEditor, swOpenWave

swOpenMsgFile(path$) As Long

Opens the message log file named path. Any existing file of the same name will be overwritten.
Returns success/fail.
See also: swCloseMsgFile, swWriteMsgFile

swOpenWave(edid&, path$) As Long

Loads the sound file specified by path into the specified editor. The editor must already be open.
Returns success/fail.
See also: swDeleteWave, swOpenEditor

swPlay(edid&) As Long

Plays the entire wave. swStopPlay does not need to be called after this, and the script will continue, without waiting for playback to finish.
Returns success/fail.
See also: swPlayAffect, swStopPlay

swPlayAffect(edid&) As Long

Plays the affect portion of the wave. swStopPlay does not need to be called after this, and the script will continue, without waiting for playback to finish.
Returns success/fail.
See also: swPlay, swStopPlay

swReceive(edid&, wavenum&, layernum&, instnum&, channel&, flags&, devicein&, deviceout&) as Long

Receives a sound via MIDI. wavenum, layernum, and instnum, are dependent on the sampler in use. Standard sample dump devices use only wavenum and ignore the other two. (see the documentation on Samplers for details). channel is the MIDI channel. flags is 0 or:

SW_SR_NOMSGS	supress message boxes

devicein and deviceout are the device numbers of the MIDI drivers to use (as presented in the Receive dialog, starting at 0).
Returns success/fail.

Example: Receive the wave in a sampler at wave number 3 using MIDI channel 5, into Wrench editor 2.

Use the first devices in the driver lists.

```
    r = swReceive( 2, 3, 0, 0, 5, 0, 0, 0 )
```

See also: swSend, swSetSamplerType

swRedraw(edid&) As Long

Redraws the waveform in the editor.
Returns success/fail.
See also: swZoomScroll

swResetPointer(edid&) As Long

Used to change the mouse pointer back to its normal look after a call to swSetWaitPointer
Returns success/fail.
See also: See also swCleanUpEditor, swSetWaitPointer

swSaveAsWave(edid&, path$) As Long

Saves the wave using the name specified by path.
Returns success/fail.
See also: swSaveWave, swSaveSplit
Example: Save the wave in editor 2 to c:\sounds\burp.wav

```
    r = swSaveAsWave( 2, "c:\sounds\burp.wav" )
```

swSaveSplit(edid&, path$) As Long

Saves a stereo wave as dual mono. The letters L and R are appended to the name path.
Returns success/fail.
See also: swSaveAsWave, swSaveWave

swSaveWave(edid&) As Long

Saves the wave using the present file name.
Returns success/fail.
See also: swSaveAsWave, swSaveSplit, swDeleteWave

swSend(edid&, wavenum&, layernum&, instnum&, channel&, flags&, devicein&, deviceout&) As Long

Sends a sound via MIDI. wavenum, layernum, and instnum, are dependent on the sampler in use. Standard sample dump devices use only wavenum and ignore the other two. (see the documentation on Samplers for details). channel is the MIDI channel. flags is 0 or the addition of any of the following:

SW_SR_CREATEINST	create a new instrument (for EPS series)
SW_SR_CREATELAYER	as above
SW_SR_CREATEWAVE	as above
SW_SR_NOMSGS	supress message boxes

devicein and deviceout are the device numbers of the MIDI drivers to use (as presented in the Receive dialog, starting at 0).
Returns success/fail.
Example: Send the wave in editor 1 to a sampler at wave number 3 using MIDI channel 5. Supress message windows and use the first devices in the driver lists.

```
r = swSend( 1, 3, 0, 0, 5, SW_SR_NOMSGS, 0, 0 )
```
See also: swReceive, swSetSamplerType

swSetAffectType(x&) As Long

Sets the Affect type. x may be:
SW_AFFECT_ALL
SW_AFFECT_INVIEW
SW_AFFECT_MARKERS
SW_AFFECT_MOUSE
Returns prior Affect type.
See also: swSetEditChannel
Example: Set the edit Affect area to everything, and save the prior value to origaffect.
```
origaffect = swSetAffectType( SW_AFFECT_ALL )
```

swSetEditChannel(x&) As Long

Sets the Edit channel. x may be:
SW_EDIT_LEFT
SW_EDIT_LEFTRIGHT
SW_EDIT_RIGHT
Returns prior Edit channel.
See also: swSetAffectType

swSetFormatType(x&) As Long

Sets the file save Format. x may be:
SW_FORMAT_8SVX
SW_FORMAT_AIFF16
SW_FORMAT_AIFF24
SW_FORMAT_AU
SW_FORMAT_RAW16I
SW_FORMAT_RAW16M
SW_FORMAT_RAW8S
SW_FORMAT_RAW8U
SW_FORMAT_RAWALAW
SW_FORMAT_RAWULAW
SW_FORMAT_REALAUDIO
SW_FORMAT_S16_3
SW_FORMAT_SD1
SW_FORMAT_VOC
SW_FORMAT_WAV32
SW_FORMAT_WAV24
SW_FORMAT_WAV16
SW_FORMAT_WAV8
Returns prior Format.
Example: Switch the save format to AIFF, save the wave in editor 1, and then switch the format back to its original value.
```
origformat = swSetFormatType( SW_FORMAT_AIFF16 )
r = swSaveWave( 1 )
swSetFormatType( origformat )
```
See also: swInquire

swSetLoop(edid&, lid&, lstart&, lend&, ltype&, nsr&) As Long

Sets the Loop to the specified positions (in samples) and type. If the specified Loop ID does not exist, a new Loop will be created. ltype may be any of the following:
SW_SETLOOP_FORWARD
SW_SETLOOP_FORWARDBACKWARD
nsr indicates whether this loop should be treated as a normal, sustain, or release loop, and may be any of the following:
SW_SETLOOP_NONE
SW_SETLOOP_RELEASE
SW_SETLOOP_SUSTAIN
Returns success/fail.
Example: For editor 1, make loop 3 the sustain loop, starting at 1000 and ending at 55000. It should be an ordinary forward-only type of loop.

```
        r = swSetLoop( 1, 3, 1000, 55000, SW_SETLOOP_FORWARD, SW_SETLOOP_SUSTAIN )
```

See also: swSetMarker

swSetMarker(edid&, mrkid&, posi&) As Long

Sets the Marker to the specified position (in samples). If the specified Marker ID does not exist, a new Marker will be created.
Returns success/fail.
See also: swSetLoop

swSetMode(edid&, mode&) As Long

Sets the Edit Mode. x may be:
SW_MODE_FREEHAND
SW_MODE_NORMAL
SW_MODE_SCRUB
SW_MODE_ZOOMBOX
Returns prior Edit Mode.

swSetMsgType(x&) As Long

Sets the output type for "OK" style messages

SW_MSGTYPE_BOX	use standard message box
SW_MSGTYPE_FILE	use log file
SW_MSGTYPE_NONE	ignore
SW_MSGTYPE_STATUS	use bottom status bar

Returns prior Message type.
See also: swCloseMsgFile, swOpenMsgFile

swSetOffset(edid&, offset&) As Long

Sets the specified axis offset for the wave. The value is in samples.
Returns success/fail.

swSetProgressBar(ByVal val&) As Long

Displays the progress bar in the right-most portion of Wrench's main status bar. The desired position is expressed as a percentage, where 0 draws an empty progress bar and 100 draws a fully extended progress bar.
Returns success/fail.
See also: swResetPointer, swSetStatusMsg, swSetWaitPointer

swSetSamplerType(x&) As Long

Sets the Sampler type. x may be:
SW_SAMPLER_AKAIS612
SW_SAMPLER_ASR10
SW_SAMPLER_DSS1
SW_SAMPLER_DSM1
SW_SAMPLER_EPS
SW_SAMPLER_EPS16P
SW_SAMPLER_P2000
SW_SAMPLER_SDS12
SW_SAMPLER_SDS16
SW_SAMPLER_SMDI
Returns prior Sampler type.
See also: swReceive, swSend, swSetFormatType

swSetSampleValue(edid&, offset&, val%) As Long
swSetSampleValueR(edid&, offset&, val%) As Long

Overwrites the existing waveform value at offset with val. offset is the sample position which ranges from 0 through the wave's size - 1. swSetSampleValueR is similar, but overwrites the value from the right channel (if one exists). val is a 16 bit integer which ranges from -32768 up to +32767.
Returns success/failure.
See also: swGetSampleValue, swInquire

swSetSampleValueFloat(edid&, offset&, val!) As Long
swSetSampleValueRFloat(edid&, offset&, val!) As Long

Overwrites the existing waveform value at offset with val. offset is the sample position which ranges from 0 through the wave's size - 1. swSetSampleValueRFloat is similar, but overwrites the value from the right channel (if one exists). val is a 32 bit single precision float which ranges from -1.0 to +1.0.
Returns success/failure.
See also: swGetSampleValueFloat, swInquire

swSetStatusMsg(msg$) As Long

Displays the passed message string in the left-most portion of Wrench's main status bar.
Returns success/fail.
See also: swSetProgressBar, swResetPointer, swSetWaitPointer

swSetWaitPointer(edid&) As Long

Changes the mouse pointer into an hourglass. Use this prior to lengthy operations. When the operation is

done, call swResetPointer
Returns success/fail.
See also: swResetPointer

swShowFull(edid&) As Long

Zooms out to show the entire wave.
Returns success/fail.
See also: swRedraw, swZoomScroll

swStopPlay() As Long

Halts any playback from within Wrench.
Returns success/fail.
See also: swPlay, swPlayAffect

swUndoWave(edid&) As Long

Undoes the wave (assuming Backups are enabled).
Returns success/fail.

swWriteMsgFile(msg$) As Long

Sends the text specified by msg to the log file.
Returns success/fail.
Example: Write the message "Now performing a violin concerto using only cheeseburgers" to the log file.
```
        r = swWriteMsgFile( "Now performing a violin concerto using only cheeseburgers" )
```
See also: swCloseMsgFile, swOpenMsgFile

swZoomScroll(edid&, zsid&) As Long

Zooms in/out or scrolls the wave. zsid may be any of the following:
SW_ZOOMSCROLL_DOWN
SW_ZOOMSCROLL_INHORIZ
SW_ZOOMSCROLL_INVERT
SW_ZOOMSCROLL_LEFT
SW_ZOOMSCROLL_OUTHORIZ
SW_ZOOMSCROLL_OUTVERT
SW_ZOOMSCROLL_RIGHT
SW_ZOOMSCROLL_UP
Returns success/fail.
Example: For editor 1, zoom in vertically.
```
        r = swZoomScroll( 1, SW_ZOOMSCROLL_INVERT )
```
See also: swRedraw, swShowFull

Sample Wrench V5 Enable Scripting

DSP Functions and Effects

Unless otherwise specified, all items in this section act upon the defined edit Affect area and the selected edit Channel(s). For details on typical values and uses, see the main DSP Functions and Effects documentation. The variable 'r' in the examples is the return value which can be checked for success/fail.

swAppend(edid&, edid2&) As Long

Inserts the wave in editor two after the end editor one's wave.
Returns success/fail.
See also: swCombine

swAM(edid&, speed#, depth#, posi&) As Long

Amplitude Modulation function. speed is the modulation sweep speed in seconds (.001 to 10), depth is the modulation depth in percent (1 to 99), and posi indicates the initial direction of the sweep. If posi is 1, the sweep will move positive and if posi is 0, the sweep will move negative.
Returns success/fail.
Example: For editor 2, create very deep modulation at 500 Hz (1/500 = .002 seconds) which starts off positive.

```
r = swAM( 2, .002, 95, 1 )
```

swChorus(edid&, delay#, fdbk#, loud#, speed#, depth#) As Long

Chorus effect. delay is time delay in milliseconds from 10 to 60, fdbk is the feedback (regeneration) percentage from 0 to 99, loud is the wet mix loudness from 1 to 100, speed is the sweep speed in seconds from .1 to 50, and depth is the sweep depth from 1 to 99 percent.
Returns success/fail.
Example: For editor 1, produce a medium delay slow chorus, which is very thick (high feedback), but with little variation (depth=10) and moderate loudness.

```
r = swChorus( 1, 30, 90, 50, 40, 10 )
```

swConvolution(edid&, convolvorid&, convolvorpath$) As Long

Convolution effect. convolvorid is the number of the editor whose wave is to be used as the convolution signal. In this case set convolvorpath to "none". If an external file is to be used instead, set convolvorid to 0, and set convolvorpath to the name of the desired external file.
Returns success/fail.
Example: For editor 1, convolve using the wave in editor 2.

```
r = swConvolution( 1, 2, "none" )
```

Example: For editor 1, convolve using the file C:\sounds\conv.wav.

```
r = swConvolution( 1, 0, "C:\sounds\conv.wav" )
```

See also: swImpulseModeling

swClickAndPop(edid&, detection&) As Long

Click and Pop Removal function. detection is one of:
SW_CLICKANDPOP_AGGRESSIVE
SW_CLICKANDPOP_NORMAL
SW_CLICKANDPOP_CONSERVATIVE
Returns success/fail.
Example: For editor 1, remove clicks and pops gently.

```
r = swClickAndPop( 1, SW_CLICKANDPOP_CONSERVATIVE )
```

See also: swNoiseReduction

swCutKeepList(edid&, process&) As Long

Processes wave chunks defined by marker pairs. process is one of:

SW_CUTKEEPLIST_CLIP	save chunks to Multi-Clipboard
SW_CUTKEEPLIST_FILE	save chunks to files
SW_CUTKEEPLIST_CUT	remove chunks, creating a new wave

Returns success/fail.

swClone(edid&) As Long

Opens a new editor and creates a copy of the original editor's waveform in it.
Returns success/fail.

swCombine(edid&, edid2&, offset&, mix&) As Long

Adds the wave in editor two to editor one's wave. offset is the point at which wave two is added and mix is the percentage of wave one (from 1 to 99 percent). The percentage of wave two will be 100 - mix (eg, a mix of 40 means 40 percent wave one and 60 percent wave two).
Returns success/fail.
See also: swAppend

swCompress(edid&, thres#, rat#, atk#, rel#, det&) As Long

Compressor/Limiter/Expandor function. thres is the threshold level in dB from -60 to 0, rat is the compression ratio from .2 to 10, atk is the attack time in milliseconds from .01 to 20, rel is the release time in seconds from .01 to 2, and det is the detection value from 1 to 50.
Returns success/fail.

swCrossMultiply(edid&, edid2&, who&, usesign&, scale&) As Long

Cross Multiply function. edid2 is the wave to multiply by. who indicates whether or not to use the present active clip rather than edid2 (1 means use clip, 0 means use edid2), if usesign is 1 then signed multiplication is used, otherwise the absolute value is used, and scale indicates the scaling factor from 0 to 6.
Returns success/fail.

swDCOffset(edid&, dc_off&) As Long

Adds or subtracts a DC (direct current) offset to the waveform. dc_off is expressed as a 16 bit quantization value. It may range from approximately -32768 (full negative scale) to +32767 (full positive scale). When working with percent of maximum, simply multiply the precent value by 327.67 to obtain the quantization value. A value of 0 will auto-detect any DC offset which is present and then remove it.
Returns success/fail.

swDeleteAffectArea(edid&) As Long

Eliminates the portion of the wave in the edit Affect area.
Returns success/fail.

swDifferentiate(edid&) As Long

Finds the rate of change (ie, slope) of the waveform in a continuous manner. This is the inverse of swIntegrate.
Returns success/fail.
See also: swIntegrate

swEcho(edid&, delay#, fdbk#, loud#) As Long

Echo function. delay is the time delay in seconds, fdbk is the feedback (regeneration) percentage from 0 to 99, and loud is the wet mix loudness percentage from 1 to 100.
Returns success/fail.
Example: For editor 1, produce many echoes spaced 1/2 second apart at a low loudness.

```
r = swEcho( 1, 0.5, 75, 20 )
```

swEnvelopeGenerator(edid&) As Long

Envelope Generator function. Applies the present envelope to the wave.
Returns success/fail.
See also: swLoadEnvGenPreset

swEQBass(edid&, hz&, db#) As Long

Bass shelving control. hz is the boost/cut hinge frequency in Hertz, and db is the cut/boost amount in decibels (-20 to +20).
Returns success/fail.
Example: For editor 1, produce a 6.5 dB boost below approximately 200 Hz.

```
r = swEQBass( 1, 200, 6.5 )
```

swEQGraFreq(edid&, hzA&, dbA#, hzB&, dbB#, hzC&, dbC#, hzD&, dbD#, hzE&, dbE#) As Long

Graphic equalizer with five bands, A through E. dbA through dbE are the cut/boost amounts in decibels (-20 to +20). hzA through hzE are the center frequencies for each band in Hertz. Set the db value to 0.0 if a band is not needed.
Returns success/fail.

swEQHighPass(edid&, hz&, order&) As Long

Removes low frequency content. hz is the frequency in Hertz below which the signal should be reduced, and order indicates the slope or steepness of the attenuation curve (1, 2, or 5).
Returns success/fail.
Example: Moderately attenuate signals below 50 Hertz in editor 3.

```
r = swEQHighPass( 3, 50, 2 )
```

swEQLowPass(edid&, hz&, order&) As Long

Removes high frequency content. hz is the frequency in Hertz above which the signal should be reduced, and order indicates the slope or steepness of the attenuation curve (1, 2, or 5).
Returns success/fail.
Example: Lightly attenuate signals above 10 kHz in the default editor.

```
r = swEQLowPass( 0, 10000, 1 )
```

swEQParametric(edid&, hz&, db#, octaves#) As Long

Parametric equalizer. hz is the center frequency in Hertz, db is the boost/cut amount in decibels, and octaves indicates the width of the cut/boost area (.1 to 3.0).
Returns success/fail.

swEQTreble(edid&, hz&, db#) As Long

Treble shelving control. hz is the boost/cut hinge frequency in Hertz, and db is the cut/boost amount in decibels (-20 to +20).
Returns success/fail.
Example: For editor 1, produce a 13 dB cut above approximately 3 kHz.

```
r = swEQTreble( 1, 3000, -13 )
```

swFFT(edid&, startoff&, records&, ptsize&, path$) As Long

Performs a Fast Fourier Transform analysis and saves it to the file specified by path. startoff is the sample offset where the analysis starts. records is the number of records to produce and ptsize is the point size of the records. ptsize may be any of 32, 64, 128, 256, 512, or 1024. The resulting file is a simple text file containing two columns. The first column is frequency in Hertz, and the second column is the relative amplitude at that frequency.
Returns success/fail.
Example: Perform a 1024 point FFT on editor 5, writing 10 records from the very beginning, and saving the result to c:\fft.dat

```
r = swFFT( 5, 0, 10, 1024, "c:\fft.dat" )
```

swFlange(edid&, delay#, fdbk#, loud#, speed#, depth#, inv&) As Long

Flange function. delay is time delay in milliseconds from 1 to 10, fdbk is the feedback (regeneration) percentage from 0 to 99, loud is the wet mix loudness from 1 to 100, speed is the sweep speed in seconds from .1 to 50, depth is the sweep depth from 1 to 99 percent, and inv indicates whether or not to invert the

signal where 1 means invert and 0 means don't invert.
Returns success/fail.

swFM(edid&, speed#, depth#, posi&) As Long

Frequency Modulation function. speed is the modulation sweep speed in seconds (.001 to 10), depth is the modulation depth in percent (1 to 99), and posi indicates the initial direction of the sweep. If posi is 1, the sweep will move positive and if posi is 0, the sweep will move negative.
Returns success/fail.

swGain(edid&, db#) As Long

Increases or decreases the amplitude of the sound. db is the boost or cut in decibels and may be upto +/- 30 dB.
Returns success/fail.
Example: For the default editor, produce a 10 dB cut.

```
r = swGain( 0, -10 )
```

swGenerate(edid&, waveshape&, rate&, freq#, dur#, dutycycleperc&) As Long

Generates a new wave. Note that the editor must be empty for this to succeed. waveshape may be any of:
SW_GEN_PULSE
SW_GEN_SAWTOOTH
SW_GEN_SILENCE
SW_GEN_SINE
SW_GEN_SQUARE
SW_GEN_TRIANGLE
SW_GEN_WHITENOISE
rate is the sample rate in Hertz (typically 11025, 22050, or 44100), freq is the frequency of the wave in Hertz (ignored for SILENCE and WHITENOISE, and which must be less than rate/2), dur is the time duration of the new wave in seconds, and dutycycleperc is the percent duty cycle from 1 to 99 (ignored for all shapes except PULSE).
Returns success/fail.
Example: Generate a 1 kHz sine wave which is 2 seconds long at a sampling rate of 44.1 kHz in editor 3.

```
r = swGenerate( 3, SW_GEN_SINE, 44100, 1000, 2, 0 )
```

swGrunge(edid&, hiss&, poplevel&, popdensity&, bitdepth&) As Long

Grunge effect. hiss is the hiss level, from 0 to 100. poplevel is the volume of pops and clicks, from 0 to 100. popdensity sets how frequent the pops and clicks occur, from 0 to 100. bitdepth is the resulting bit truncation length. Valid bitdepths are 6, 8, 10, 12, and 14. To ignore bit depth reduction, use 0 for bitdepth.
Returns success/fail.

swHarmony(edid&, sourcelevel&, vox1pitch&, vox1level&, vox2pitch&, vox2level&) As Long

Harmony effect. sourcelevel is the volume of the original wave. vox1level and vox2level are the volumes for the first and second added voices. vox1pitch and vox2pitch are the shifts for the first and second added voices. levels are in percent, from 0 to 100. pitches are in cents, from -2400 to +2400. Set level to 0 if a voice is not needed.

Returns success/fail.
Example: For editor 2, add one voice one octave (+1200 cents) above the original. Reduce original volume to 70% and make the new voice 60% in volume.

```
r = swHarmony( 2, 70, 1200, 60, 0, 0 )
```

See also: swPitchShift, swResynthesize

swImpulseModeling(edid&, impulseid&, impulsepath$, startms&, endms&, shiftms&, mix&, reverse&, fadeout&, stereolink&) As Long

ImpulseModeling effect. impulseid is the number of the editor whose wave is to be used as the impulse signal. In this case set impulsepath to "none". If an external file is to be used instead, set impulseid to 0, and set impulsepath to the name of the desired external file. startms and endms specify the number of milliseconds to ignore at the beginning and ending of the impulse. shiftms is the amount the impulse is moved forward or backward in time, again in milliseconds. Positve value of shiftms delay the impulse will negative values make it occur earlier. The final four items are checkmarks for optional processing and must be set to 0 for unchecked or 1 for checked. When checked (set to 1) they mean:

mix	Combine the effect with the current wave rather than replacing it.
reverse	Flip the impulse backwards
fadeout	Apply a decreasing gain to the impulse
stereolink	Crossfeed the two channels for an alternate style of reverb

Returns success/fail.
See also: swConvolution

swInsertSilence(edid&, amount#, units&, where&) As Long
swInsertSilenceAt(edid&, amount#, units&, where&) As Long

Inserts a silent chunk at the specified position.
amount is the size of the silent chunk to be added, with units specified by one of:
SW_INSERT_SECONDS
SW_INSERT_MILLISECONDS
SW_INSERT_SAMPLEPOINTS
For swInsertSilence, where is one of:
SW_INSERT_SAMPLESTART
SW_INSERT_SAMPLEEND
SW_INSERT_AFFECTSTART
SW_INSERT_AFFECTEND
For swInsertSilenceAt, where is the insertion position in sample points.
Returns success/fail.
Example: For editor 1, add 3 seconds of silence at sample offset 1000.

```
r = swInsertSilenceAt( 1, 3.0, SW_INSERT_SECONDS, 1000 )
```

swIntegrate(edid&) As Long

Finds the area under the curve in a continuous manner. This is the inverse of swDifferentiate.
Returns success/fail.
See also: swDifferentiate

swInvert(edid&) As Long

Flips the wave upside down so that the positive portion is now negative, and vice versa.
Returns success/fail.

swMaximize(edid&) As Long

Increases the signal to its maximum level (just prior to clipping).
Returns success/fail.

swMonoStereo(edid&, edid2&) As Long

Turns a mono sound into a stereo one. edid2 is the editor containing the wave which will become the right channel for editor one. If wave two is longer than wave one, wave two will be truncated. If wave two is shorter than wave one, wave two will be padded with silence at the end.
Returns success/fail.
See also: swStereoMono

swMute(edid&) As Long

Reduces the sound to zero volume.
Returns success/fail.

swNoiseGate(edid&, thres#, atk#, rel#, det&) As Long

Noise Gate function. thres is the threshold level in dB from -10 to -90, atk is the attack time in milliseconds from .01 to 20, rel is the release time in seconds from .01 to 2, and det is the detection value from 1 to 50.
Returns success/fail.

swNoiseReduction(edid&, process&, noiseprintid&, noiseprintpath$, amount&, tracking&, threshold#, thresholdtype&) As Long

Background Noise Reduction function. noiseprintid is the number of the editor whose wave is to be used as the noiseprint signal. In this case set noiseprintpath to "none". If an external file is to be used instead, set noiseprintid to 0, and set noiseprintpath to the name of the desired external file.
amount indicates the overall severity of processing, and is one of:
SW_NOISEREDUCTION_AMOUNTHEAVY
SW_NOISEREDUCTION_AMOUNTNORMAL
SW_NOISEREDUCTION_AMOUNTLIGHT
tracking indicates whether or not the signal is quickly changing, and is one of:
SW_NOISEREDUCTION_TRACKINGRAPID
SW_NOISEREDUCTION_TRACKINGNORMAL
SW_NOISEREDUCTION_TRACKINGSTEADY
process is one of:
SW_NOISEREDUCTION_PROCESSNONE
SW_NOISEREDUCTION_PROCESSTHRESHOLD
SW_NOISEREDUCTION_PROCESSNOISEPRINT
SW_NOISEREDUCTION_PROCESSBOTH
thresholdtype is either:
SW_NOISEREDUCTION_THRESHSTATIC
SW_NOISEREDUCTION_THRESHDYNAMIC
threshold is the level below which frequency components are suppressed, from -100 dB to -20 dB.
See also: swClickAndPop

swNormalize(edid&, db#) As Long

Increases or decreases the signal to the desired level. db is the resultant peak signal relative to clipping. A range of 0 to -40 is allowed.
Returns success/fail.
Example: For the default editor, set peak to 10 dB below clipping.

```
r = swNormalize( 0, -10 )
```

swPitchShift(edid&, shift#, xbass&, stereolink&) As Long

Changes pitch without changing time. shift is the shift factor from -400 to +400 cents, xbass indicates the presence of low frequency content (1 if extended bass response, 0 if not), and stereolink indicates whether phase-locked processing should be used (1 if phase-locked, 0 if not).
Returns success/fail.
See also: swResynthesize, swTimeStretch, swHarmony

swRectifyFull(edid&) As Long

Fullwave rectifies the sound (flips negative portions up so that they're positive).
Returns success/fail.

swRectifyHalf(edid&) As Long

Halfwave rectifies the sound (removes negative portions).
Returns success/fail.

swRemove(edid&) As Long

Alternate name for swDeleteAffectArea().

swReplicate(edid&, reps#, choice&) As Long

Creates replicas of the waveform segment. reps is the amount of replication desired. choice indicates how to interpret reps, and may be any of:

SW_REPLICATE_MILLISECS	reps = total milliseconds to add
SW_REPLICATE_REPS	reps = total repititions of the area to add
SW_REPLICATE_SECS	reps = total seconds to add

Returns success/fail.
Example: For editor 1, create 3 seconds worth of replicas.

```
r = swReplicate( 1, 3.0, SW_REPLICATE_SECS )
```

swResynthesize(edid&, id&, timetype&, freqtype&, discrimtype&, timepitch#) As Long

Resynthesis function. id indicates the type of resynthesis to perform and may be either:

SW_RESYNTH_PITCH	treat timepitch as a pitch shift value
SW_RESYNTH_TIME	treat timepitch as a time shift value

The next group sets the operation, speed and accuracy of the resynthesis. timetype may be any of:
SW_RESYNTH_TIMEFAST
SW_RESYNTH_TIMENORMAL
SW_RESYNTH_TIMESHORT
freqtype may be any of:
SW_RESYNTH_FREQHIGH
SW_RESYNTH_FREQLOW
SW_RESYNTH_FREQNORMAL
discrimtype may be any of:
SW_RESYNTH_DISCRIMFAST
SW_RESYNTH_DISCRIMNORMAL
SW_RESYNTH_DISCRIMSLOW
timepitch is the amount of shift. For pitch the range is -2400 to +2400 cents, and for time the range is .25 to 4.0.

Returns success/fail.
See also: swPitchShift, swTimeStretch, swHarmony

swReverb(edid&, time#, pre#, damp#, loud#, diff#, enc&) As Long

Reverberation function. time is the reverb decay time in seconds (.2 to 20), pre is pre-delay time in milliseconds (0 to 200), damping is the high frequency damping (0 to 20 where higher means less high frequency content), loud is the wet mix loudness percentage (1 to 100), diffusion is the diffusion factor (0 to 20), and enc indicates the enclosure which may be any of:
SW_REVERB_HALL
SW_REVERB_ROOM
SW_REVERB_SPRING
Returns success/fail.
Example: For editor 2, create an 11 second reverb with a 52 millisecond predelay. Use moderate damping, a high wet mix, low diffusion, and the ROOM enclosure.

```
r = swReverb( 2, 11, 52, 9, 80, 3, SW_REVERB_ROOM )
```

swReverse(edid&) As Long

Swaps the sample back to front so that it plays backwards.
Returns success/fail.

swSampleRateTranspose(edid&, fs&, choice&, prefil&, adj&) As Long

Transposes Sample Rates. fs is the new sample rate in Hertz, choice indicates how the process is performed and may be any of the following:

SW_SRT_LINEAR	linear interpolation
SW_SRT_RATE	don't change wave, just the rate
SW_SRT_RESAMPLE	resample

if prefil is 1 then prefilter before conversion (0 to ignore), and if adj is 1 then adjust markers and loops to achieve the same relative positions rather the same sample offsets (0 to ignore).
Returns success/fail.
Example: For editor 2, change the sample rate to 22 kHz using linear interpolation. Adjust the markers and loops but don't prefilter.

```
r = swSampleRateTranspose( 2, 22000, SW_SRT_LINEAR, 0, 1 )
```

swScaleToFull(edid&) As Long

Alternate name for swMaximize().

swSilence(edid&) As Long

Alternate name for swMute().

swSpectralWarp(edid&, startfac#, startbias#, endfac#, endbias#, logfac&, logbias&) As Long

Spectral Warp function. startfac and endfac are the starting and ending factors respectively (.1 to 10), startbias and endbias are the starting and ending bias values respectively (-1000 to 1000), if logfac is 1 then use log factor scaling (0 gets linear factor scaling), and if logbias is 1 then use log bias scaling (0 gets linear bias scaling).
Returns success/fail.

swStereoMono(edid&, choice&) As Long

Converts a stereo sound to a mono one. choice is one of:

SW_SM_L	left only
SW_SM_LMR	left minus right
SW_SM_LPR	left plus right
SW_SM_R	right only
SW_SM_SWAP	swap right and left

Returns success/fail.
See also: swMonoStereo

swTimeStretch(edid&, stretch#, xbass&, stereolink&) As Long

Changes time without changing pitch. stretch is the stretch percentage from 75 to 125, xbass indicates the presence of low frequency content (1 if extended bass response, 0 if not)), and stereolink indicates whether phase-locked processing should be used (1 if phase-locked, 0 if not).
Returns success/fail.
Example: For the default editor, increase the length by 10 percent (ie, 110 percent), use extended bass response, and don't bother with stereo linking.

```
        r = swTimeStretch( 0, 110, 1, 0 )
```

See also: swPitchShift, swResynthesize

swTransferFunction(edid&) As Long

Transfer Function. Applies the present transfer curve to the wave.
Returns success/fail.
See also: swLoadTransFuncPreset

swTrim(edid&) As Long

Removes areas outside of the edit Affect area.
Returns success/fail.

swUnClip(edid&) As Long

Attempts to smooth clipped regions of a wave.
Returns success/fail.

Sample Wrench V5 Enable Scripting

Example Scripts

These scripts are complete and self-contained. Use them as is, modify them for your own needs, or use them as starting points or modules for larger scripts.

```
' version.bas
' This script displays the present version info
'
Sub Main ()
	m = swGetVersionInfo()
	MsgBox "The version is: " & m
End Sub
```

```
' active.bas
' Displays current active editor
'
Sub Main ()
	id = swGetActiveEditorID()
	MsgBox "The active editor ID is: " & id
End Sub
```

```
' affect.bas
' Displays the current edit Affect type
'
Sub Main ()
	m = swSetAffectType( SW_AFFECT_ALL )
	' Change it back to what we had
	swSetAffectType( m )
	' Determine and print type
	Select Case m
	Case SW_AFFECT_ALL
		msg = "Affect All"
	Case SW_AFFECT_INVIEW
		msg = "In View"
	Case SW_AFFECT_MARKERS
		msg = "Markers"
	Case SW_AFFECT_MOUSE
		msg = "Mouse"
	Case Else
		msg = "Error"
	End Select
	MsgBox "The edit Affect type is: " & msg
End Sub
```

```
' attack.bas
' This Script adds some "attack" or "bite" to a sound by producing
' a modest boost of 3dB to the upper midrange frequencies. This
' works on the wave loaded into the default (active) editor.
Sub Main ()
        x = swEQParametric( 0, 3500, 3.0, 1.0 )
End Sub
```

```
' markers.bas
' This script shows how to define and edit arbitrary regions of a sound.
' First, it defines two markers (0 and 1) at locations 2000 and 5000,
' respectively. The edit Affect area is then changed to MARKERS type. A
' simple edit is made (Silence) and the edit Affect type is changed back
' to its original form. This is performed on the default (active) editor.
Sub Main ()
        ' Make the Markers used by the MARKERS type (this redefines them if
        ' they already exist)
        x = swSetMarker( 0, 0, 2000 )
        If x = 0 Then Return
        x = swSetMarker( 0, 1, 5000 )
        If x = 0 Then Return
        origaffect = swSetAffectType( SW_AFFECT_MARKERS )
        ' Silence the area between the two markers
        swSilence( 0 )
        ' Further edits can be made here or the two markers can be redefined
        ' for new edit areas like so:
        ' x = swSetMarker( 0, 1, 4233 ) ' change marker 1 to position 4233
        swSetAffectType( origaffect )
End Sub
```

```
' eqhp2.bas
' This script is an example of creating an alternate interface to an existing
' Wrench function. This presents a dialog with a text box, and OK and Cancel
' buttons. The value in the text box will be used as the frequency for a
' second order high pass filter. The default (active) editor is used.
'
Sub Main ()
        Begin Dialog AltFilt 20,20, 150, 60, "2nd Order High Pass"
                Text 10,10, 28,12, "Freq:"
                TextBox 40,10, 100,12, .hertz
                OKButton 10,30, 40,12
                CancelButton 100,30, 40,12
        End Dialog
        Dim dlg1 As AltFilt
        dlg1.hertz = 95      ' default value is 95 Hertz
        ' Dialog returns -1 for OK, 0 for Cancel
        button = Dialog( dlg1 )
        If button = 0 Then Return
        ' Turn the string of characters into a double
        f = CDbl( dlg1.hertz )
        x = swEQHighPass( 0, f, 2 )
End Sub
```

```
' convert.bas
' Open an editor and load a wave called c:\sounds\borzoi.svx into it.
' Scale the wave to full size, change the save format to 16 bit WAVE,
' and save the sound as c:\sounds\borzoi.wav
Sub Main ()
        id = swOpenEditor()
        If id = 0 Then Return        ' couldn't open editor
        x = swOpenWave( id, "c:\sounds\borzoi.svx" )
        If x = 0 Then Return         ' couldn't load the wave
        swScaleToFull( id )
        origform = swSetFormatType( SW_FORMAT_WAV16 )
        x = swSaveAsWave( id, "c:\sounds\borzoi.wav" )
        swSetFormatType( origform )
End Sub


' batchconv.bas
' This script is an example of batch file conversion with processing.
' Change save format to 8 bit WAVE, open an editor, load a wave,
' move its sample rate to 22050 Hz if it isn't already, remove
' most frequency content below 75 Hz, and save it. Repeat for a series
' of waves
Sub Main ()
        Dim innames$(4)
        innames(0) = "c:\source\aardvark.aif"
        innames(1) = "c:\source\bunny.svx"
        innames(2) = "c:\source\cow.sd1"
        innames(3) = "c:\source\dingo.aif"
        Dim outnames$(4)
        outnames(0) = "c:\dest\aardvark.wav"
        outnames(1) = "c:\dest\bunny.wav"
        outnames(2) = "c:\dest\cow.wav"
        outnames(3) = "c:\dest\dingo.wav"

        id = swOpenEditor()
        If id = 0 Then Return        ' couldn't open editor
        origform = swSetFormatType( SW_FORMAT_WAV8 )

        For sound = 0 to 3
                x = swOpenWave( id, innames(sound) )
                If x <> 0 Then
                        rate = swInquire( id, SW_INQUIRE_RATE, 0 )
                        If rate <> 22050 Then
                                x = swSampleRateTranspose( id, 22050, SW_SRT_LINEAR, 1, 0 )
                        End If
                        x = swEQHighPass( id, 75, 2 )
                        x = swSaveAsWave( id, outnames(sound) )
                End If
                ' Clean up for next time
                swDeleteWave( id )
        Next sound
        swCloseEditor( id )
        swSetFormatType( origform )
End Sub


' batchconv2.bas
' This script is an example of batch file conversion with processing.
' Unlike batchconv.bas, this version does everything in-line without
' loops, so each sound can have specific processing.
Sub Main ()
        id = swOpenEditor()
        If id = 0 Then Return        ' couldn't open editor

        ' Open up gorgo, change rate to 11025 Hz, and save as WAV 8 bit
        ' under the new name ourgon.wav
        origform = swSetFormatType( SW_FORMAT_WAV8 )
        x = swOpenWave( id, "c:\sounds\gorgo.aif" )
        If x <> 0 Then
                rate = swInquire( id, SW_INQUIRE_RATE, 0 )
                If rate <> 11025 Then
                        x = swSampleRateTranspose( id, 11025, SW_SRT_LINEAR, 1, 0 )
                End If
                x = swSaveAsWave( id, "c:\out\ourgon.wav" )
        End If
        ' Clean up for next time
        swDeleteWave( id )

        ' open up kiwi, low pass filter it at 10 kHz, and save as WAV 16 bit
        x = swSetFormatType( SW_FORMAT_WAV16 )
        x = swOpenWave( id, "c:\sounds\kiwi.svx" )
        If x <> 0 Then
                x = swEQLowPass( id, 10000, 2 )
                x = swSaveAsWave( id, "c:\out\kiwi.wav" )
        End If
        ' Clean up for next time
        swDeleteWave( id )

        swCloseEditor( id )
        swSetFormatType( origform )
End Sub


' batchdir.bas
' This script is an example of batch file conversion for an entire directory.
' This converts all files in the directory C:\audio\old that end in .voc
' to 8 bit wav format, saving them in the directory C:\audio\new
Sub Main ()
        olddir = "C:\audio\old\"
        newdir = "C:\audio\new\"

        id = swOpenEditor()
        If id = 0 Then Return        ' couldn't open editor
        origform = swSetFormatType( SW_FORMAT_WAV8 )
        f = swFindFirstFile("C:\audio\old\*.voc")

        While f <> ""
                x = swOpenWave( id, olddir & f )
                If x <> 0 Then
                        ' trim off the old extension, and replace it with .wav
                        l = Len(f)
                        f = Left$(f,l-3)
                        x = swSaveAsWave( id, newdir & f & "wav" )
                End If
                ' Clean up for next time
                swDeleteWave( id )
                f = swFindNextFile()
        Wend

        x = swCloseFindFile()
        swCloseEditor( id )
        swSetFormatType( origform )
End Sub



' findfile.bas
' A simple example of using swFindFirstFile, swFindNextFile,
' and swCloseFindFile to search through a directory to find
' the names of available files. In this example, all files
' ending in ".aif" in the directory "C:\audio" are found.
' These file names can then be used to load the files for
' batch processing. swFindFirstFile MUST be called first.
' If it returns an empty string then no files match the
' target search string. Otherwise, it returns the name of
' the first matching file it finds. After this, calls to
' swFindNextFile return other matching files. When swFindNextFile
' returns an empty string, there are no more matching files.
' Finally, call swCloseFindFile when you are finished so that
' Sample Wrench can do required clean-up.
'
' Another way to search for specific extensions is to extract
' it using this technique:
' ext = UCase$(Right$(f,3))
' and then perform appropriate comparisons. This is useful if
' you want to match a couple of different extensions using a
' single loop.
'
Sub Main ()
        x = 1     'a simple counter
        f = swFindFirstFile("c:\audio\*.aif")

        ' keep going as long as we didn't get an empty string
        ' and display the filename in a window
        while f <> ""
                MsgBox "File " & x & " is: " & f
                x = x + 1
                f = swFindNextFile()
        wend
        ' clean up
        x = swCloseFindFile()
End Sub


' batchdump.bas
' This script is an example of sample dump batch file.
' Sounds are brought in and then saved as 16 bit WAV files.
Sub Main ()
        Dim outnames$(4)
        outnames(0) = "c:\dest\aardvark.wav"
        outnames(1) = "c:\dest\bunny.wav"
        outnames(2) = "c:\dest\cow.wav"
        outnames(3) = "c:\dest\dingo.wav"

        origtype = swSetSamplertype( SW_SAMPLER_SMDI )
        origform = swSetFormatType( SW_FORMAT_WAV16 )

        id = swOpenEditor()
        If id = 0 Then Return       ' couldn't open editor

        For sound = 0 to 3
                ' the template is swReceive( editor, sound, SCSI device, LUN, host adapter, 0, 0 )
                x = swReceive( id, sound, 2, 0, 0, 0, 0 )
                If x <> 0 Then
                        x = swSaveAsWave( id, outnames(sound) )
                End If
                ' Clean up for next time
                swDeleteWave( id )
        Next sound

        swCloseEditor( id )
        swSetFormatType( origform )
End Sub


' msgstatus.bas
' Changes "OK" messages in Wrench from window box style to status bar style.
'
Sub Main ()
        swSetMsgType( SW_MSGTYPE_STATUS )
End Sub


' logfile.bas
' Shows how to use the log file.
'
Sub Main ()
        ' While we're at it, redirect any "OK" style messages to the log file too
        swSetMsgType( SW_MSGTYPE_FILE )
        swOpenMsgFile( "c:\logit.txt" )
        swWriteMsgFile( "Doing scale to full on the default editor." )
        swScaleToFull( 0 )
```

```
		' Do similar function calls as the above two...
		swCloseMsgFile
		' Be safe: turn the message style back to window boxes
		swSetMsgType( SW_MSGTYPE_BOX )
End Sub


' reversbe.bas
' This is a neato effet script which creates reverse reverb (ie, the
' reverberation comes before the sound which creates it, creating an
' interesting swelling effect. This is done by reversing the sound,
' applying reverb, and then reversing it again. The sound itself is
' reversed twice so it comes out normal, but the reverb is reversed
' only once, so it comes out backwards.
' This works on the default editor (0).
Sub Main ()
		x=swReverse(0)
		'Create a 2 second reverb with a 25 millisecond predelay.
		'Use moderate damping, a high wet mix, low diffusion,
		'and the ROOM enclosure.
		x=swReverb( 0, 2, 25, 10, 90, 4, SW_REVERB_ROOM )
		x=swReverse(0)
End Sub


' rampy.bas
' This script creates a custom sound sample. It is 2 seconds long and looks like a
' cross between a triangle wave and a ramp. Note that no error checking is performed
' to verify success/fail in order to keep this as simple as possible. Also, the default
' editor is chosen (0).
Sub Main ()
		' First, create 2 seconds worth of silence
		x=swGenerate(0, SW_GEN_SILENCE, 22050, 1000, 2, 0)
		' Find out how long the sample is in sample points
		l=swInquire(0,SW_INQUIRE_SIZE,0)
		' Initialize the value (v) to 0 so there isn't a starting pop.
		v = 0
		inc = 200
		' This may take a moment so put up the hourglass.
		swSetWaitPointer(0)
		' Cycle through the sound sample by sample. Note that the positive increment
		' is smaller than the negative going increment, thus the positive slope is
		' slower than the negative slope.
		for i = 0 to l
				x=swSetSampleValue(0,i,v)
				v = v + inc
				' If we've hit the limits, reverse direction.
				if v > 31000 then inc = -1000
				if v < -31000 then inc = 200
		next i
		' We've changed the data so redraw the waveform.
		swRedraw(0)
		' Get back the original mouse pointer.
		swResetPointer(0)
```

```
End Sub


' rampy2.bas
' This script creates a custom sound sample. Here, a dialog box is
' used in order to obtain the sample rate and length of the sound.
' This represents the tip of the iceberg in creating your own
' personal Sample Wrench "plug-ins".
'
Sub Main ()
	Begin Dialog Rampy 20,20, 150, 80, "Rampy Wave Maker"
		Text 10,10, 28,12, "Rate:"
		TextBox 40,10, 100,12, .rate
		Text 10,30, 28,12, "Secs:"
		TextBox 40,30, 100,12, .len
		OKButton 10,50, 40,12
		CancelButton 100,50, 40,12
	End Dialog
	Dim dlg1 As Rampy
	dlg1.rate = 22050	' default value is 22050 Hertz
	dlg1.len = 1		' default length is 1 second
	' Dialog returns -1 for OK, 0 for Cancel
	button = Dialog( dlg1 )
	If button = 0 Then Return
	' Turn the strings of characters into doubles
	samprate = CDbl( dlg1.rate )
	samplen = CDbl( dlg1.len )

	x=swGenerate(0, SW_GEN_SILENCE, samprate, 1000, samplen, 0)
	l=swInquire(0,SW_INQUIRE_SIZE,0)
	v = 0
	inc = 200
	swSetWaitPointer(0)
	for i = 0 to l
		x=swSetSampleValue(0,i,v)
		v = v + inc
		if v > 31000 then inc = -1000
		if v < -31000 then inc = 200
	next i
	swRedraw(0)
	swResetPointer(0)
End Sub


' slappy.bas
'
' This script creates simple slap echoes from scratch,
' without using Wrench's swEcho() function. The purpose here is
' show just how sample data can be directly manipulated via Enable.
' A custom dialog is used in order to obtain the desired length of
' the slap echo (short=50 millisec, med=75 millisec, long=100 millisec).
' The echo is created by delaying the signal the required time and
' then adding the delayed signal back in (at half of its signal level
' so it's not too obtrusive). The delay itself is created by placing
' the sound data into an array and pulling it out later (this is
' called a circular buffer).
'
' This also shows the proper usage of the Status Message and
' Progress Bar calls.
'
' This effect operates on the entire sound sample. It does not check
' for edit Affect areas or channels (mono/left only).
Sub Main ()
        Dim delaypts As Long
        Dim samplepts As Long
        Dim samplerate As Long
        Dim i As Long
        Dim iindex As Long
        Dim oindex As Long
        Begin Dialog Slappy 0,0, 172, 92, "Slap Echo Generator"
                Text 16,12,68,12, "Select Duration:"
                OptionGroup .GRP
                OptionButton 12,32,80,12, "Short ( 50 mSec )"
                OptionButton 12,48,80,12, "Medium ( 75 mSec )"
                OptionButton 12,64,80,12, "Long ( 100 mSec )"
                OKButton 108,28,48,12
                CancelButton 108,60,48,12
        End Dialog
        Dim dlg1 As Slappy
        dlg1.GRP = 1        'default to medium

        samplepts=swInquire(0,SW_INQUIRE_SIZE,0)
        If samplepts = 0 Then
                MsgBox "There is no wave here!"
                Return
        End If
        samplerate=swInquire(0,SW_INQUIRE_RATE,0)

        ' Dialog returns -1 for OK, 0 for Cancel
        button = Dialog( dlg1 )
        If button = 0 Then Return
        ' determine which option was chosen and then compute size of delay in points
        Select Case dlg1.GRP
                Case 0
                        sec = .05
                Case 2
                        sec = .1
                Case Else
                        sec = .075
        End Select
        delaypts = Int( samplerate * sec + .5 )

        ' this can take a moment, so put up the hourglass pointer and such
        swSetWaitPointer(0)
        swSetStatusMsg("Enable Slappy script running")

        ' create the delay buffer and initialize in/out index
        ' note: Enable does not support dynamic arrays- the one below
        ' is big enough for .1 Sec delay at 100 kHz sampling rate
        Dim buf (10000) As Integer
        iindex = 1
        oindex = 2
        lastp = 0

        ' this is the DSP loop
        For i = 0 To samplepts Step 1
                v = swGetSampleValue(0,i)
                buf(iindex) = v
                v = v + buf(oindex)/2
                x = swSetSampleValue(0,i,v)

                ' check for index overflow, and increment
                If iindex >= delaypts Then
                        iindex = 1
                Else
                        iindex = iindex+1
                End If

                If oindex >= delaypts Then
                        oindex = 1
                Else
                        oindex = oindex+1
                End If

                ' update progress bar
                ' this technique updates every 1% instead of every
                ' sample point in order to keep overhead down
                p = Int( 100 * i / samplepts )
                If p <> lastp Then
                        swSetProgressBar(p)
                        lastp = p
                End If
        Next i

        ' reset visuals and such

        ' swRedraw(0)
        ' swSetProgressBar(0)
        ' swSetStatusMsg("Ready")
        ' swResetPointer(0)
        ' the four calls above (and more) are accomplished with the following:
        swCleanUpEditor(0)
End Sub


' preecho.bas
'
' This script creates a pre-echo from scratch, without using
' Wrench's swEcho() function. This is similar to the slappy.bas script,
' but the echo here occurs before the sound, rather than after.
' A custom dialog is used in order to obtain the desired length of
' the echo (short=50 millisec, med=75 millisec, long=100 millisec).
' The echo is created by adding the present value to whatever is found
' at the delay length later. No buffer is required.
'
' This also shows the proper usage of the Status Message, Backup, and
' Progress Bar calls, along with the preferred method of finishing
' via CleanUpEditor.
'
' This effect uses the presently defined edit Affect area and channels.
Sub Main ()
        Dim delaypts As Long
        Dim samplepts As Long
        Dim samplerate As Long
        Dim i As Long
        Dim samplestart As Long
        Dim sampleend As Long
        Begin Dialog Slappy 0,0, 172, 92, "Pre-Echo Generator"
                Text 16,12,68,12, "Select Duration:"
                OptionGroup .GRP
                OptionButton 12,32,80,12, "Short ( 50 mSec )"
                OptionButton 12,48,80,12, "Medium ( 75 mSec )"
                OptionButton 12,64,80,12, "Long ( 100 mSec )"
                OKButton 108,28,48,12
                CancelButton 108,60,48,12
        End Dialog
        Dim dlg1 As Slappy
        dlg1.GRP = 1    'default to medium

        samplepts = swInquire(0,SW_INQUIRE_SIZE,0)
        channels = swInquire(0,SW_INQUIRE_CHANNELS,0)
        If samplepts = 0 OR channels = 0 Then
                MsgBox "There is no wave here!"
                Return
        End If
        samplerate = swInquire(0,SW_INQUIRE_RATE,0)

        samplestart = swInquire(0,SW_INQUIRE_AFFECTSTART,0)
        sampleend = swInquire(0,SW_INQUIRE_AFFECTEND,0)
        doleft = swInquire(0,SW_INQUIRE_AFFECTLEFT,0)
        doright = swInquire(0,SW_INQUIRE_AFFECTRIGHT,0)

        ' Dialog returns -1 for OK, 0 for Cancel
        button = Dialog( dlg1 )
        If button = 0 Then Return
        ' determine which option was chosen and then compute size of delay in points
        Select Case dlg1.GRP
                Case 0
                        sec = .05
                Case 2
                        sec = .1
                Case Else
                        sec = .075
        End Select
        delaypts = Int( samplerate * sec + .5 )
        ' this can take a moment, so put up the hourglass pointer and such
        swSetWaitPointer(0)
        swSetStatusMsg("Enable PreEcho script running")
        swBackupEditor(0)

        lastp = 0
        samplepts = sampleend - samplestart + 1
        ' this is the DSP loop
        For i = samplestart To sampleend Step 1
                If doleft = 1 Then
                        v = swGetSampleValue(0,i)
                        ' no need to check for overrun on the index since Wrench
                        ' will return 0 if we're out of bounds
                        d = swGetSampleValue(0,i+delaypts)
                        v = v + d/2
                        x = swSetSampleValue(0,i,v)
                End If

                If doright = 1 Then
                        v = swGetSampleValueR(0,i)
                        d = swGetSampleValueR(0,i+delaypts)
                        v = v + d/2
                        x = swSetSampleValueR(0,i,v)
                End If

                ' update progress bar
                ' this technique updates every 1% instead of every
                ' sample point in order to keep overhead down
                p = Int( 100 * (i-samplestart) / samplepts )
                If p <> lastp Then
                        swSetProgressBar(p)
                        lastp = p
                End If
        Next i

        ' reset visuals and such
        swCleanUpEditor(0)
End Sub
```