Virtual Wind Instrument Design Guidelines

R. D. Villwock

3-3-09

Revised
5-11-12

Table Of Contents

1.0 Introduction

Wind instruments are among the most difficult instruments to synthesize in a realistic manner. One of the obstacles to achieving realism (for sample-based synthesis) is the ‘shop-worn’ piano paradigm on which most sample-based instruments are patterned. Using key-down velocity to control dynamics is not too well-suited for control of wind instruments. The piano paradigm works out well for many percussive and plucked instruments but, for many other instruments (including bowed strings), a different approach is sorely needed. **I want to emphasize that most of the ideas in this guide do not originate with me**, rather they have been introduced by several pioneering developers. I’m just trying to collect them together in one place.

Wind instruments can be very expressive and a lot of that expressiveness arises from the way a performer uses his lungs, lips and tongue to control air flow through the instrument. In an effort to impart this kind of control to a synthesizer, several kinds of wind-operated MIDI controllers have been developed. While these controllers have the **potential** to provide a more convincing emulation for wind instruments, they cannot achieve this potential unless the **Virtual Instrument** (**VI)** they are ‘connected’ to is specifically designed with these wind controllers in mind. Among other things, this means that the instrument designer must be familiar with the technical operation and limitations of wind controllers.

Since this document is aimed at **VI** design, the focus will be confined to ‘emulative’ rather than ‘innovative’ synthesis. One purpose of this document is to point out some of the shortcomings of the piano model (with or without wind control) for wind-instrument synthesis, and, to discuss other strategies which can achieve more realistic results (especially with wind-driven controllers). A second purpose is to present some basic information about the MIDI output of wind-operated controllers (under various conditions) and, to suggest some of the ways for a **VI** to interface with these controllers to achieve more realistic emulations. Finally, it should be mentioned that the ideas presented herein will assume a soft-synth environment like that of Kontakt which includes a very capable script processing feature.

2.0 Wind Instruments Versus the Piano

We want to discuss here a few of the most important differences between a wind instrument and a piano insofar as it impacts the design of a **VI**. When you play a key on a piano, the key specifies which note you want to sound. And the force, with which you strike it, determines how loud you want the note to be played. After this initial ‘bang’ on the strings, the sound produced decays over time in a mostly uncontrolled way (apart from using the damper pedal). The MIDI specification was pretty much developed around this piano keyboard control philosophy in that the note-on message carries both the desired pitch and a velocity value to indicate the ‘initial force’ of the attack. It’s not too surprising then that most synths and samplers are also designed with this same ‘piano philosophy’ in mind.

But now, let’s compare how a wind instrument is played against how a piano is played. Pressing a key on a wind instrument, in and of itself, produces no sound. In order to produce a sound, you must blow into the instrument. If you blow hard, the note that sounds will be loud and if you blow softly, the note will be softer. However, while you are sustaining the note, you can raise or lower its volume by raising or lowering the amount of air pressure you are continuing to provide. The note doesn’t stop sounding until you completely stop the flow of air.

Besides these rather obvious facts about wind instruments, there is another fact that may not be so obvious at first thought. Wind instruments do not have a ‘no note’ key fingering. i.e. **When you blow into a wind instrument even with all the keys or valves released, a note will still sound**. Unlike a piano where ‘all keys released’ means **no note**, ‘all keys released’ on a wind instrument is just one of its many key combinations. All such combinations specify a note (or notes) which will be played when air is blown into the instrument. This distinction should also be remembered when designing a **VI** interface for use with wind-driven controllers.

3.0 A Paradigm for Virtual Wind Instruments

Since velocity data is generated only once at the beginning of each note, **it should not be used to control volume dynamics** (like it would be for a piano-type instrument). However, since velocity is available with each played note, we should try to put it to good use. Some of the more obvious things we might use velocity for include controlling the sharpness and tuning of a note's attack and/or the legato connectivity (possibly including portamento effects at low velocity levels).

Volume dynamics, for a virtual wind instrument, should be controlled from something that can act during the entire sustain of a note (just as with the real instrument). Since volume can change during the sustain of any given note, it means the velocity-layer idea needs to be re-thought. Instead of calling them velocity layers, perhaps we should refer to them as **volume layers**. After all, their purpose is to alter the timbre as the volume level is changed. However, if the volume level is not controlled by velocity, then the name velocity layer is a misnomer. See **Section 5.0** for a more detailed discussion of the wind instrument paradigm and see **Section 5.2** in particular for more on the use of volume layers.

4.0 Wind-Driven Controllers

In addition to keyboard control, a virtual wind instrument should also provide a musically useful interface for wind-driven controllers. So, before we discuss any more details of the control paradigm to use for virtual wind instruments, we should first describe the kind of wind-driven controllers that are currently available and what can be expected from them.

There are basically two categories of wind-driven controllers. When a wind-driven controller contains a set of keys and is a MIDI **instrument** unto itself (in that it can be used **without** any support from a synth keyboard), we will refer to it as a **Wind Controller** or **WC** for short. When a wind-driven controller is simply used as an adjunct to a synth keyboard, we will use the name **Breath Controller**, or **BC** for short. Since both these controllers produce an output related to blowing into them, they obviously have some points of similarity, but, they also have notable differences which should be considered when configuring a **VI** for use with either or both of these controller types.

4.1 Breath Controllers

As defined in this document, a **BC** (such as the Yamaha BC3) has only one MIDI controller output (usually assigned to CC2) which is proportional to the air pressure that results from blowing into it. Two electrical adjustments are provided with the BC3. One is the **Offset** and the other is the **Gain**. The **Offset** setting determines the minimum threshold pressure required before **any** MIDI output is generated and, the **Gain** setting determines how much harder you have to blow to cause the MIDI output to swing from the threshold value to full scale. In addition, there is a mechanical adjustment that regulates the amount of 'back pressure' by controlling the 'leak rate' of a small valve. Since a **BC** has only a wind-pressure MIDI output, **BC**'s are usually used in conjunction with a keyboard (with the keyboard providing the usual note-on/off, velocity, and possibly aftertouch control) and the **BC** providing the dynamics control.

4.2 Wind Controllers

A **WC** (such as the Akai EWI) also produces a MIDI controller output related to air pressure, but in addition, the mouthpiece contains a 'bite sensor' that can be assigned to a different MIDI controller. Usually, the bite sensor is assigned as an AC-coupled Pitch Bender and can thus be used to produce vibrato effects directly. In addition the **WC** contains a set of finger-actuated keys to provide note-on/off events. The **WC** is usually played in a manner similar to that of a flute or saxophone, although some of the EWIs (as well as other, specialized wind controllers) also support Valve-Instrument fingering schemes. In addition, some **WC**s provide additional touch sensors such as 'bender plates' to allow for portamento effects. Because a **WC** can provide note-on/off control as well as breath dynamics, the **WC** (unlike the **BC**) can be used without a synth keyboard if the **VI** is configured to accommodate it.

4.3 Wind Controllers Versus Breath Controllers

From a purely technical point of view, it makes little difference whether the 'fingering' is performed on a synth keyboard or with a **WC**. The additional realism is provided primarily by the wind-pressure to MIDI-controller function and this ability is common to both the **WC** and the **BC** so there is no special 'realism' advantage in using a **WC**. In fact, using a **BC** with a keyboard has several advantages over using a **WC**. This is due to the way a **WC** generates its velocity data and to the **WC**'s lack of articulation control (see sections **4.4** and **6.0**). However, there are also some practical considerations that may tilt the scale in favor of one type over the other, so let's briefly discuss some of the pros and cons of **WC** versus **BC**.

If you are a skilled wind instrument player but not too good with keyboards, using a **WC** might make sense for you. But, keep in mind that using a **WC** actually involves the effort of learning to play another musical instrument, even though it may be rather similar (but certainly not identical) to an instrument you already play. However, learning to play a **WC** also has the possible advantage of giving you another instrument (with a very broad pallet of sounds) for live playing.

On the other hand, if you are a skilled keyboard player with little or no wind instrument chops, you may prefer to play the notes on your synth keyboard and just handle the dynamics with a suitable MIDI CC. For this you might use your **Mod Wheel** or you might add an **Expression Pedal** to your keyboard rig (if you don't already have one). However, some tonguing sounds made by wind instruments are difficult to simulate with a **Mod Wheel** or an **Expression Pedal**. But, if you add a **BC** to your keyboard rig,. with very little practice (compared to learning to play a **WC**), you will be able to generate dynamics control in a way much more like the real acoustic instrument you are trying to emulate. While the average keyboard player may not have much experience using a **BC**, not all keyboard players routinely use an expression pedal either. And, it really isn't any more difficult to learn to use a **BC** than to learn to use an **Expression Pedal.** Learning to use a **WC** on the other hand will require considerably more effort, especially if you have no prior wind instrument skill.

4.4 MIDI Output of Wind Controllers

A detached note is played on a **WC** by first positioning the fingers on the key combination that will provide the desired pitch and then blowing into the mouthpiece. This causes a note-on message to be generated. And, the velocity of the note-on will usually be proportional to the air pressure (or in some cases the rate of change of air pressure). At the same time, the MIDI CC assigned to breath (usually CC2) will also reflect the air pressure rise at the start of the note as well as its level or variation throughout the sustain. When the air flow is stopped, CC2 will of course drop below threshold but in addition, a note-off message will be generated. The 'threshold' being referred to here is determined within the **WC** itself and may or may not be user adjustable. But like the BC3, most **WC**s provide a **Gain** adjustment to set the full-scale sensitivity.

For legato playing, when the note fingering is changed (while still maintaining the air flow above threshold), a new note-on will be generated and this will be followed quickly (usually within a ms or so) by a note-off message for the prior note. The velocity of the new note-on will again be proportional to the air pressure when the new note is started. Therefore, there are two conditions under which a **WC** will generate a note-off message. The first is when the air pressure drops below threshold (as for the end of a detached note or the end of a legato phrase). The second is when an overlapping note is played. Overlapping notes are created with a **WC** when the note fingering is changed while maintaining the air pressure above threshold. Any time the **WC** generates such a new note-on event, it will automatically issue a note-off (shortly thereafter) for any prior note that was sounding. This behavior is similar to the Mono or Solo mode offered by some synths.

Some **WC**s (for example the USB EWI) can optionally be set to generate a fixed velocity value for each note-on (independent of the current air pressure). But, a more common practice is that the velocity of each note-on generated by the **WC** is dynamically related to the air pressure at the time of the note-on. As a result, it is rather difficult to control velocity independently from the attack dynamics with a **WC**. On the other hand, when using a synth keyboard and **BC** combination, velocity can be controlled in a completely independent way from the breath dynamics. This is another important distinction between **WC** and **BC** that **VI** developers need to consider when deciding how to utilize velocity and what kind of options to provide for various modes.

5.0 Other Design Details

In **Section 3.0**, a general virtual wind instrument model was introduced. The purpose of this section is to present an expanded and more detailed view of that model and to introduce some additional design guidelines that might prove useful. But, before we even get going, let's discuss the emotionally-charged topic of Reverb.

5.1 To Reverb or Not To Reverb

One often hotly debated topic is whether or not sampling should be done in some acoustically-beautiful room. The realism that can be achieved with a **VI** is directly proportional to the ability it has to manipulate a small number of static samples in realistic-sounding ways. You can never provide enough articulations and variations to emulate every nuance of any given musical instrument by 'direct recording' of all these nuances, especially those of wind instruments.

If you insist on recording your samples in a 'live' room, you will greatly limit the kinds of things you can do later to give these samples real musical life. If you want your instrument or library to have the sound of some beautiful space, take advantage of convolution technology. Carefully and precisely record a high-quality **Impulse Response** of the room sound you want and then record your samples in a dry space. With any reasonable-quality convolution reverb, if you have recorded the room's **IR** properly, you will be able to add the room after-the-fact with little or no sonic difference. Whatever small loss you might incur, will be more than overshadowed by the wonderful benefit of flexible sample manipulation and processing that you can add *before* re-creating the room. Once the room is made part of the sample, a ton of flexibility goes out the window. Some of the most-realistic sounding virtual instruments today were recorded in an anechoic chamber (and rooms don't get any deader than that).

5.2 Volume Dynamics Control

As already discussed in **Section 3.0**, volume dynamics should be controlled from some MIDI CC that will enable volume to be changed at any time during the sustain of a note. The preferred dynamics controller will of course be a **BC** or **WC** (usually assigned to CC2) but, lacking any kind of wind-driven controller, the next best choice for dynamics control would probably be an **Expression Pedal** (usually assigned to CC11). Although not the best choice, the **Mod Wheel** is often used since just about every user will have one. Therefore, the most flexible design will be to make the dynamics CC a user-selectable option.

As the dynamics controller is varied from soft to loud, we will of course want the volume (of a played or sustained note) to increase from soft to loud. However, we will also want the timbre to change in a realistic way. Smooth, realistic-sounding continuous volume/timbre control during a note's sustain turns out to be one of the more challenging departures from the piano paradigm. So far, the problem has been attacked from two distinct directions. The 'modeling' method attempts to use some form of dynamic equalization to darken the sound for lower volume playback. The 'sampling' method attempts to utilize the traditional velocity layers in some way to provide a smooth morph over the instrument's dynamic range. Both of these approaches have their strong and weak points.

While sampling inherently provides more realism, trying to morph smoothly between discrete static samples is problematic. Usually, sample morphing is accomplished by crossfading a number of discrete samples but this often produces a distinct phasing sound in the overlap regions. This more or less forces the performer to pass through these overlap regions quickly to ‘avoid hearing it’ (which then often warps the intended musical dynamics). There are some techniques for getting rid of (or at least subduing) these phasing problems but most of them require very arduous processing of the **volume-layer** samples. Probably the benchmark for phase-free morphing of samples is a proprietary process introduced by Sample Modeling which they refer to as ‘Harmonic Alignment’. But most devleopers have neither the facility nor the know-how to accomplish something similar and as a result, phase-free morphing of volume layers remains somewhat of an elusive dream.

Dynamic EQ on the other hand can provide very smooth continuous morphing but at the expense of less realism. Modern samplers have fairly powerful EQ facilities and the right combination of settings can produce a pretty convincing effect, so don’t dismiss this idea out of hand. Just consider the following list of benefits and you’ll realize that this technique has a lot going for it.

1. Absolutely no phasing problems
2. No added polyphony
3. Greatly reduced sample count
4. Reduced memory usage and load time.
5. Easy to setup and easy to adjust (even after the fact).

Using only the FF samples, one EQ scheme that seems to work quite well (and produces very musical results) is to use a single-section, parametric EQ with the frequency set around 5KHz to 8KHz. Use a fairly low Q (ie wider bandwidth) and arrange it so that the EQ provides flat response when the dynamics controller is at max. As the dynamics controller is moved toward min, the EQ should be set to proportionally increase the EQ’s attenuation at the center frequency. You will probably also need to reduce the normal amplifier volume with the dynamics controller to get the right combination of darkening timbre and reduced volume.

Better yet, if you are using Kontakt 4/5 and you have multiple volume-level samples for your instrument, you should consider using the AET filter. This is a very sophisticated form of dynamic EQ that essentially tunes the EQ over the morphing range to accurately match the spectral characteristics of the lower-volume samples (analyzed by Kontakt as a separate process). It’s not perfect and there may be stubborn samples for which it doesn’t perform as well as we might like but, generally, the AET can do an amazing job of simulating a smooth morph in timbre over an instruments dynamic range.

If you’ve never tried using the AET Filter, you may want to get a copy of a Tutorial that I wrote titled: **Using Kontakt’s AET Filter**. Also, David Carpenter is planning a series of tutorials related to virtual wind instruments. Among other things, David plans to discuss some of his techniques for getting the most from the AET Filter. David has also been exploring such things as breath/wind controller conditioning in Kontakt, using convolution to emulate timbre variations (such as trumpet mutes), etc. Watch for his tutorials when then appear because I’m sure they will contain a lot of valuable information to help you construct more realistic Virtual Wind Instruments.

5.3 Optional Modes

A virtual wind instrument should be designed to operate in 3 basic modes. These are **Keyboard Only**, **Breath Controller**, and **Wind Controller** or **KO**, **BC**, and **WC** for short. In the **KO** mode, the default dynamics controller should be CC11 but this should be user re-assignable. In **BC** and **WC** modes the default dynamics controller should be CC2 (but again user re-assignable). The default dynamics controller is often the only difference between the 3 modes (in which case **BC** and **WC** modes would be identical). However, a well-designed **VI** will also benefit from the more-subtle differences between these 3 controllers.

5.3.1 Keyboard Only Mode

Since keyboards are inherently polyphonic and wind instruments aren't, the scripting should be designed to provide **Solo-Mode** control so that when overlapping notes are played, the latest note-on should force the prior note-off within a few msec in a manner similar to that produced by a **WC** (see 4.4). Another important scripting feature is the handling of the sustain pedal for legato-connected notes. When the sustain pedal is down, detached notes should be played as if they were overlapped. This should not generate chords but rather it should insure that legato connections will occur for otherwise detached notes. One of the important side-effects of implementing the sustain pedal this way is that it allows the performer to play **legato-connected notes of the same pitch**. This is used a lot in jazz venues and yet, most **VI**'s do not provide any way to accomplish this in **KO** mode.

Besides dynamics control (via an expression pedal), one should take advantage of the fact that keydown velocity can be controlled and utilized independently from the dynamics control and therefore can be put to other uses. The most common way velocity can be used is to alter the hardness of the attack for detached notes and, to control the legato effect for overlapped notes. If the legato effect is scripted (similar to SIPS Legato for example), one can assign shorter crossfade and bend times to the higher velocities and vice versa. Also, below some low velocity threshold, the legato transition can be morphed into a portamento glide (if this is appropriate for the instrument being emulated).

The Solo-Mode logic should also include a re-triggering of prior notes (if their keys are still down). This is sometimes called a **trill feature** and basically the idea is that when a pair of notes are played overlapped (ie as a legato interval), if the first note's key is still down when the 2nd key is released, the script will re-trigger the first note. This makes it very easy to play realistic-sounding trills and even things like trumpet shakes.

The **VI** should also respond to Pitch Bend messages and allow control of vibrato intensity with the Mod Wheel and/or channel aftertouch. The **VI** should also provide realtime user control of a number of instrument-specific parameters via MIDI CCs to add additional touches of realism approriate to a given instrument.

5.3.2 Keyboard Plus Breath Controller Mode

For this mode, all the note fingering and such is still done with the keyboard and therefore, any special effects controlled by keyboard velocity can be inherited from the **KO** mode. But the availability of a **BC** can enable a more realistic emulation of wind instruments. Certain kinds of tonguing, not to mention sforzandos and such, can be very difficult if not impossible to perform realistically with just an expression pedal whereas these effects are fairly easy to perform with a **BC**.

While note fingering is handled by the keyboard, note-ons should be triggered only when both a key is down **and BC** air flow is present. The note-off condition should occur when either the key is lifted **or** the air flow drops below threshold. Moreover, if a phrase is played with overlapping keyboard notes while the **BC** remains above threshold, each new note played should force a note-off for the prior note within a few milliseconds **after** the new note-on occurs (in a manner similar to that of a **WC**, see 4.4)

Another difference between a synth keyboard and a real wind instrument is that there is no counterpart to all-keys-up. Wind instruments have no key combination that represents ‘no note’, that is, when you blow into a wind instrument it always sounds some note no matter what keys are down or up. However, to mimic this behavior precisely in the **BC** mode might be very counterintuitive for a keyboard player. Therefore, I think it might be best to not generate a note-on when the **BC** is above threshold but no keys are down. However, if a note-off occurs due to stopping the **BC** airflow (while a key is still held down), raising the **BC** pressure again should create a new note-on. Thus the **BC** mode is nicely summarized as already outlined in the preceding paragraph, namely: keydown **and BC** triggers a note-on condition whereas a note-off condition is triggered when **either** the key is lifted or the **BC** falls below threshold.

5.3.3 Wind Controller Mode

At least for live playing, **WCs** are usually used without any synth keyboard support. In other words, a **WC** is a complete MIDI instrument by itself. However, most **WC**s do not provide any convenient means to change articulations dynamically. Thus, when a virtual wind instrument depends upon multiple articulations for realistic playing, a **WC** player may be at a real disadvantage. This ‘side-effect’ of the current trend in **VI** design (to include tons of articulations, usually selected with keyswitches or program change commands) can be a real problem in designing a friendly **WC** mode. This subject will be discussed a little further in **Section 6.0**.

WCs often have ‘bender’ plates and other assignable CCs such as a bite sensor. A **VI** for use with a **WC** should provide at least some optional means to enable these controllers to do something useful. For example, the bite sensor of an EWI might be set by the performer to the AC-Coupled pitch bender mode so the performer can produce real vibrato. For that to be able to work, the **VI** must be able to vary it’s pitch in response to bender messages. Conversely, if the **VI** generates its own vibrato, it should be optionally defeatable in **WC** mode (so as not to compete with the bite sensor pitch variations). An EWI can also be configured to have its bite sensor send CC1 messages. In this case, the **VI** might be expected to generate its own vibrato but allow the intensity of the effect to be changed with CC1.

Since there are differences in the way that performers like to configure a **WC**, a well-designed **VI** should provide a number of control options so the user can choose which are most comfortable for his/her playing style. The minimum function needed to support these additional controllers is a user-settable pitch-bend response but this of course is also required in both the **KO** and **BC** modes as well so accommodating a **WC** for pitch bending doesn’t require anything special. Obviously one thing that should occur when the **VI** is put in **WC** mode is the default dynamics controller should be set to CC2 (just as for **BC** mode). However, since **WC**s automatically generate their own note-on/off as well as pressure data, no special solo-mode logic is needed to support a **WC**.

One other potential problem area for a **WC** is related to how the **KO** and **BC** modes are designed to respond to velocity data. In the **KO** and **BC** modes, velocity can easily be controlled in an independent way relative to breath pressure. With a **WC**, usually there are at most two note-on velocity options. The **WC** either generates a fixed velocity value for all note-on events, or alternatively, the **WC** generates a velocity value proportional to the air pressure (CC2 value) at the time the note-on is created. What this boils down to is that the performer doesn’t have the means to vary velocity independently from attack dynamics as you would have with a keyboard plus expression pedal (or keyboard plus **BC**). With **KO** or **BC** mode, you might arrange it so that intervals played at very low velocities cause a portamento glide between notes with a slower glide time as the velocity gets lower. However, this may not work out very well with a **WC** and you may have to provide a user option to de-couple glide time from velocity

6.0 Articulation Switching

The trend with **VI**s is to include more and more articulations. This is usually done in an effort to circumvent the major weakness of sample-based synthesis. Compared with physical modeling, sample-based synthesis provides a more realistic timbre but at the expense of less realistic control. The 'frozen' sound of a sample quickly becomes obvious, often in the simplest of musical contexts.

When you listen to demos of any given **VI**, you will often find heavy use of what I'll call the 'pyrotechnics' of using many of the 'effects' articulations to more or less cover up the static frozen sound of sampling technology. Rarely will you hear a demo of a new **VI** that showcases the instrument 'out front' and playing a simple melodic line with great expression. This then becomes almost the acid test for a **VI**'s ability to produce a very realistic emulation.

Kontakt's inclusion of the KSP (Kontakt Script Processor) has opened up some new territory that allows much more realistic control of 'frozen' samples. And, in the light of this, it might be better to work on better ways to manipulate a smaller number of samples than to just keep expanding the articulation count. Some of the most realistic-sounding **VI**s so far have come from Sample Modeling and their instruments do not use dozens of articulations to achieve that realism. Rather, Sample Modeling has found ways to give the performer greater control over a smaller number of sample types.

When multiple articulations are provided, with the idea that they can be switched in and out dynamically while playing, a player-friendly way has to be provided to accomplish the articulation switching. The two most common control methods are key switching and patch-changing (using the MIDI Program Change event). When there are a lot of articulations, linear keyswitching requires a lot of keys above and/or below the normal playing range of the instrument. Users with a 61-note synth keyboard, may be dismayed to discover that not all of the keyswitches are 'on-line' at the same time and thus require octave shifting to reach. On the other hand, issuing a Program Change command from a keyboard usually involves pushing a number of buttons in rapid succession so neither of these two methods are too friendly for for live playing.

Of course for those recording to a sequencer and also willing to edit the sequence to insert or modify the articulation switching commands, possibly by making multiple passes, the choice between keyswitches, PC commands, or whatever may be of far less importance. However, many users that are recording to a sequencer are under schedule pressures also and so anything which can reduce the number of passes required to get things sounding right is of interest to them. Personally, I think a 2-tier keyswitching scheme may be the best compromise if only keyswitching or PC is considered. A 2-tier KS layout using **N** keys, allows quick access to $\mathbf{N^2/4}$ articulations. So, one octave (13 keys) will allow access to as many as 42 articulations compared with only 13 for a single-tier system. For a skilled keyboard player, it should be no more difficult to hit a pair of keys within an octave than it would be to hit one of 42 keys spread over 3.5 octaves. Again, we should take advantage of the script processor.

However, I think moving in the direction of fewer rather than more articulations is the better way to go. This of course assumes that we can use scripting to provide an even better equivalent of many articulations. If one can write a script that provides very realistic-sounding vibrato for example, it would be a superior solution to providing one or two frozen vibrato articulations. Of course if there are only a small number of articulations, the number of keyswitches is no longer the issue. However, there might be some additional articulation switching techniques that could be considered. For example, a foot switch that would sequence through each articulation. Such an option might be of great interest to **WC** players. As mentioned in **Section 5.3.3**, most **WC**s do not provide any means for articulation switching and this means that to play a **VI** with multiple keyswitched articulations, a **WC** player must either change articulations with a mouse or that he/she has to set up an auxiliary synth keyboard for this purpose. Neither of these methods is too well suited for live playing.

7.0 About WIPS

Incorporating all the foregoing strategies into a Kontakt-Based virtual wind instrument will require a combination of skillful Kontakt programming and some custom scripting. The **W**ind **I**nstrument **P**erformance **S**uite, **WIPS**, is a family of Kontakt Scripts designed to work together harmoniously to provide some of the most important functions outlined in this document. While WIPS has its ancestry in SIPS, it is nevertheless a totally fresh scripting effort that benefits from many of the KSP enhancements added since the days of K2.2. In addition, the design of WIPS takes advantage of several additional years of hindsight gained in using SIPS and in thinking about the special requirements for realistic wind instrument synthesis.

WIPS includes a very powerful and flexible articulation and variation control system with very musical-sounding legato, portamento, and vibrato effects. WIPS also supports the 3 major methods of virtual wind instrument control with **Keyboard Only**, **Keyboard + Breath Control**, and, **Standalone Wind Controller** modes. Mode 2 also supports **Keyboard + Expression Pedal** for those who don't yet have a breath control setup. All 3 modes are designed to provide the special control characteristics that are most needed by virtual wind instruments.

WIPS articulations are made up of complex combinations of Kontakt Instrument Groups and each articulation can contain an arbitrary number of 'natural' (ie sampled) variations as well as synthesized (TKT) variations. Each variation can have as many as 6 groups triggered simultaneously to support all manner of volume-layer blending, mod-wheel crossfading, etc. In addition, special support is included to make using the AET filter easier. Both normal and release groups are supported by all WIPS articulations. And, unlike SIPS, WIPS handles all articulation control without the need to alter group-start programming (which has been a troublesome area in the past and then became unusable with the introduction of K4).

Up to 100 complex articulations can be defined and 'called up' in real time by means of MIDI Controller messages and/or keyswitches. Keyswitches can be freely assigned to any articulation individually or in pairs anywhere on the keyboard. This allows for all manner of linear or bank-keyswitch configurations. Keyswitches can be either latching or non-latching in any combination. A dedicated set of variation parameters (such as triggering mode, sequencing mode, etc) are saved and recalled with each articulation rather than being set globally. In addition, Legato/Vibrato panel setups can be associated with any articulation so that legato and vibrato parameters can be changed on the fly as articulations are changed.

The improved solo-mode logic used in WIPS also supports a retrigger option for performing realistic trills, trumpet shakes, etc. The 'hold/ghost' pedal function makes it possible to perform legato-connected repeat notes and such things as starting a legato phrase with a legato attack (and bending from any desired imaginary note). Any pair of notes in a legato phrase can easily be connected with a smooth, formant-corrected glide allowing all manner of portamento effects such as emulating a slide trombone glissando. Both legato bending and glide time contours can be adjusted for different shapes ranging from linear to 2nd-order concave in the negative regions and from linear to 2nd-order convex in the positive regions of the contour control. Separate control of down and up bends are provided for added realism.

Small random variations of all the key legato parameters can be programmed for more realistic humanization of the effect and all legato and glide parameters can be CC controlled during performance using Kontakt's automation facility (extended to allow the use of **Channel Aftertouch**, **Pitch Wheel** or played **Velocity** to control legato/glide parameters during performance). In addition, the articulation script includes a powerful meta-articulation facility that will allow you to use keyswitch automation of various mode buttons.