Section 3

THE TUTORIALS

Tutorial 1: Entering Notes with the Note Editor

The *Note Editor Menu* is probably the most complex menu within the sequencer and getting the most out of it requires a little patience and a little practice. This tutorial is intended to walk you through a typical edit session and perhaps show you a couple of hints and tips that will make future sessions quicker, more productive and more enjoyable.

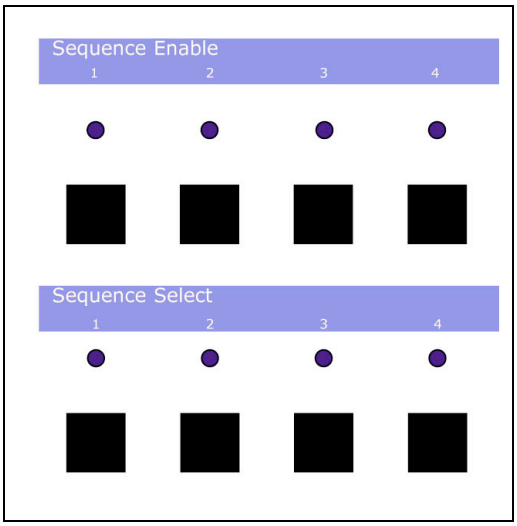
As mentioned elsewhere, there are three methods for entering note information into *ZEIT*. These are:

1) Pitch Control knobs
2) The Step Input Menu
3) System Exclusive Messages

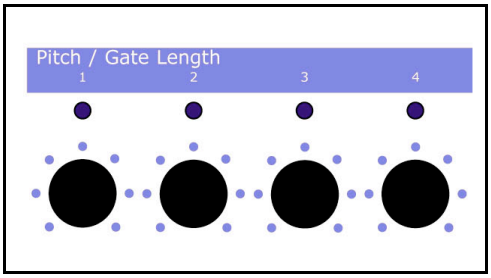
In this section, we'll look at the first two methods.

Straight out of the box, *ZEIT* contains a small number of factory preset sequences. These are not protected in any way and may be overwritten by your own sequences.

Let's start with the first sequence. Use *the Sequence Enable* push buttons to make sure that only the first sequence is enabled. Next, make sure that any changes you make using the front panel affect just this sequence by pressing the *Sequence Select* push button on the bottom right of the front panel.



Press *Play* to listen to just Sequence 1 on its own and press the *Normal/Shift* push button so that the instrument is in *Normal* mode and the LED above the *Shift* push button is switched off.



Go to the top row on the front panel and slowly turn one of the pitch knobs in the clockwise direction. *ZEIT* will notice that you've changed the pitch of one of the steps and jump directly to the *Note Editor*. The display will show the new note values as you turn the knob.

The *Note Editor* can only show the pitch details for 4 steps at the same time simply because the display isn't big enough to accommodate any more steps. As a compromise, the *Note Editor* displays steps 1 to 4 followed by steps 5 to 8 then 9 to 12 and finally, 13 to 16. You can move between the steps using the *Enter* push button.

When you change a note value, the cursor jumps to the relevant step and will sit just to the left of the pitch value. You can continue to use the pitch knob to change the note value or, equally, you can use the *Data Wheel* instead.

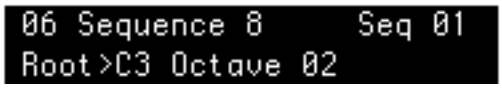


The note values that you can select will depend upon the *force-to-scale* setting for the current scale. If *force-to-scale* is switched on then you will only be able to select note pitches within the current scale. So, if the C Major scale is selected then you will only be able to select the notes C, D, E, F, G, A and B. If *force-to-scale* is switched off then you can select any note pitch i.e. C, C#, D, D# etc.

Next, try changing a couple of pitch knobs at the same time. Notice what the display does. The display will show the new pitch value for the first note you changed but won't display any of the other notes until you've stopped changing the first knob. The step values themselves will be updated and you'll hear the new note pitch when the step is next played but you won't see the note value being updated. This is for technical reasons and stops the display jumping from one page to the next as you update the notes.

The Root Note and Octave Range

By *default* the sequencer only lets you choose notes in the range C3 to C6. Suppose that you want to select a note from a lower octave, say C2. To do this, we need to change the *Root Note* parameter.



Each step in a sequence can have its own root note, however, there is a global setting in the *Sequence Play* Editor, which lets you set the root note for all 16 steps at once. This function also has a *hot-key* on the front panel in the *Pitch/MIDI* section labelled *Pitch*. Press this *hot-key* once and the cursor will point to the *Root* parameter. Use the *Data Wheel* to change the root note for all of the steps at the same time. Notice that changing this parameter does not affect the notes that the sequencer plays. This parameter only affects the notes available on the pitch knobs.

Next, notice the second parameter on this page. This is the *Octave* parameter and sets the number of octaves that the pitch knobs can cover. The lowest setting is 1 and the highest is 4. If the *Root Note* is set to C1 and the *Octave* setting is 1 then the notes available on the pitch knobs are restricted between C1 and C2. When this setting is 2 then the notes available on the pitch knobs are between C1 and C3. Why do we limit the range of notes available on the pitch knobs? The number of notes available under the MIDI specification is 128, which is over 10 octaves. Picking a specific note from such a wide range of values is difficult and so we chose to limit the range of the pitch knobs.

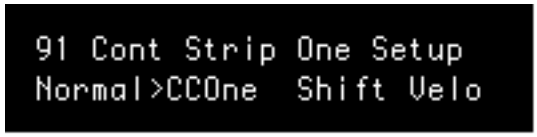
However, suppose that you want a sequence to have a mixture of notes from a number of octaves. As we said above, it's possible to set a root note for each step individually.

To do this, press the *Note Edit* push button on the menu bar and use the *Data Wheel* to scroll through the available pages until you find the *Root Note* as shown:

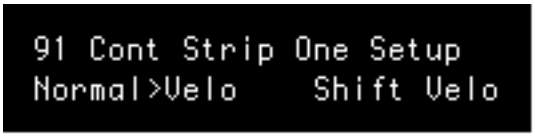


Press the *Enter* push button to step forwards through the steps. You can use the *Data Wheel* to vary the root note for each step in the sequence. This can be a bit slow and tedious and so we added another method to speed up the process.

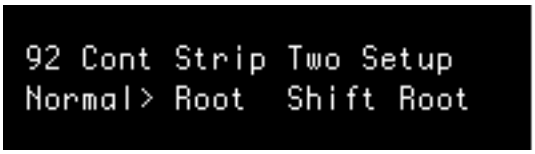
What follows depends upon how you have the *Controller Strips* defined in the *System Setup* Menu. By default they should be configured to update MIDI Continuous Controller One and MIDI Continuous Controller Two respectively in *Normal Mode* and the Note Velocity and Root Note for each step in *Shift Mode*. For the purposes of this tutorial, go to the *System* Menu and scroll through the available pages until you come to the *Controller Setup* pages.



Use the *Data Wheel* to change the above settings so that Controller Strip One affects the Note Velocity in both *Normal* and *Shift* modes.



Go to the second Controller setup page and change the settings so that *Controller Strip Two* updates the *Root Note* in both *Normal* and *Shift* modes.



Now, go to the *Controller Two* strip on the front panel and turn one of the knobs. You can now update the *Root Note* for each step directly from the front panel. Equally, go to the *Controller One* strip and turn one of those knobs. It's now possible to directly edit the *Note Velocity* for each step directly from the front panel rather than through a menu.

HINT: To move quickly from one menu to another or even to another page within the same menu, just press the relevant button on the menu quick select bar. This should save you a lot of time navigating through the various pages.

We can also input notes into a sequence using an attached MIDI keyboard. Press the *Note Editor* push button on the *menu bar* twice to return you to the *Note* page of the *Note Editor.*



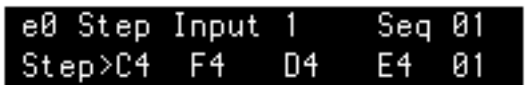
Press the *Enter* button a couple of times so that the cursor points at *Step 2.* If you have a MIDI keyboard attached to the *MIDI In* port on the rear panel, try pressing a key. The note for the current step will take on the value of the key you just pressed. The note value is unaffected by the state of the *Force-to-Scale* setting or the *Root Note* setting for the current step and/or sequence.

Equally, if you now go to the *Note Velocity* page, you will find that the *Velocity* value for the currently selected note now reflects the velocity with which you played the note. This is a quick way of changing individual notes in a sequence but you can only change one at a time and you need to step through the entire range of steps using the *Enter* push button.

Using the *Step Input* menu

Dialling in a sequence using the pitch knobs is pretty fast but nowhere near as fast as just adding them in from a keyboard! Equally, some people find it easier to input notes directly from a MIDI keyboard and then use *ZEIT's* editing facilities to make their sequences jump around.

Press the *Step Input* push button on the *Menu Bar*. The first page is the one we're interested in.



The *Step Input* section has a couple of operating modes that you can use to better match your compositional needs. These are addressed on *Page 2* of the *Step Input Menu* and we'll come to those shortly.

Press the *Enter* push button once so that the cursor points at Step 1. Next, press a note on your keyboard and watch the display. Step 1 will take on the value of the note you pressed on your keyboard and the cursor will jump to Step 2. This happens whether the sequencer is playing or not.

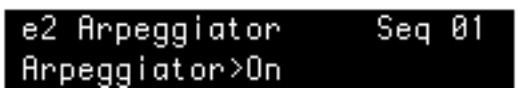
You can play as many or as few notes as you like. To skip past a note just use the *Enter* push button. When you've played all 16 notes in from the keyboard, the cursor will return to the home position. Use the *Data Wheel* to scroll to the next page.

By default the *Step Input* mode is set to *Step-Time* instead of *Real-Time*, which means that the cursor will only advance to the next step after you've pressed a key on your MIDI keyboard. In *Real-Time* mode, *ZEIT* will drop any note played on the attached keyboard on top of the current step, as defined by the step position LEDs. Use the *Data Wheel* to set the *Step Input* mode to *Real-Time* and then return to the previous page.



Move the cursor so that it points to Step 1 and then play a note on your keyboard. If the sequencer is not playing, only Step 1 will be updated. Now press *Play* in the *Transport Strip* and start playing notes on your keyboard.

Changing notes in this way takes practice and patience but can create some wonderfully dynamic sequences. However, you do need to watch your settings in the *Keyboard Setup Menu* to ensure that you don't have any conflicts.



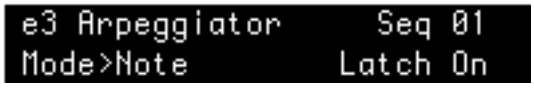
Finally, we're going to look at using the *Arpeggiator* to input a sequence. Press the *Play* push button so that playback stops and, staying within the *Step Input Menu*, scroll down to the *Arpeggiator Status page*.

Use the *Data Wheel* to set the *Arpeggiator Status* to *On* and press a couple of keys on your keyboard. You won't hear anything. This is because the sequencer must be playing for the *Arpeggiator* to work.

Okay, press *Play* so that playback begins. Any other sequences that you might have enabled will begin to play back but you should notice that the currently selected sequence will not play until you press a note. Press the *Shift* push button and notice that the length of the sequence is set to 1 step and that this step is muted.

Now play and hold a couple of notes on your keyboard. Straightaway, you should hear a rising series of notes. This is the normal *Arpeggiator* function. But how do we get *ZEIT* to use the *Arpeggiator* notes as the basis for a sequence?

To do this, go to the *Arpeggiator Setup* page, change the *Latch* setting to *On* and press *Play*.



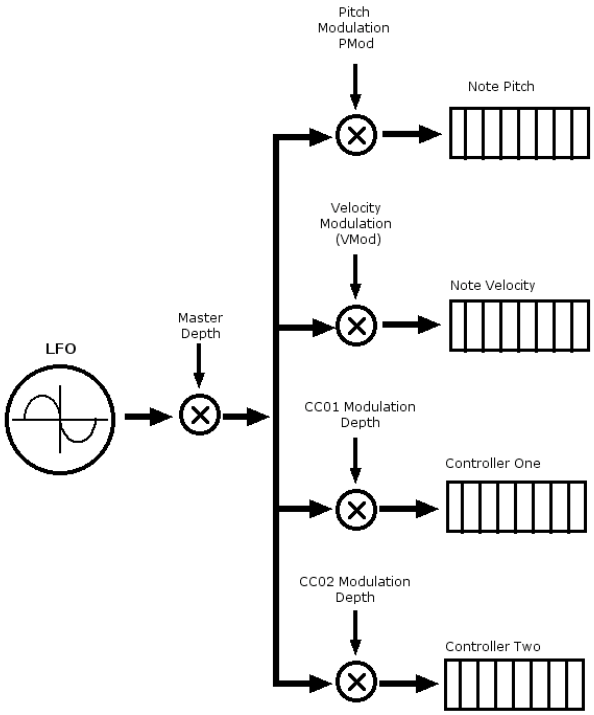
As usual, playback begins. Carefully select the notes that you want to play and *ZEIT* will add each new note to its *Arpeggiator Buffer. H*owever, this time, they will not be released when you let go of them. These notes are said to be *Latched*. To release these notes you must stop playback, mute the sequence itself or go to the *Arpeggiator Setup* page and set the *Latch* parameter to *Off.*

Latched notes are held within the *Note Edit Buffer* for the currently selected sequence and you can use the normal *Note Editor* to change their respective values.

Tutorial 2: Using the Low Frequency Oscillators

In this tutorial, we're going to look at using the Low Frequency Oscillators to introduce periodic changes to our sequences.

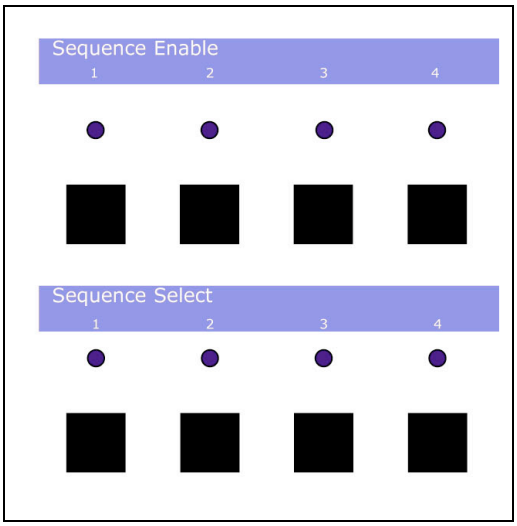
The *Low Frequency Oscillators* (*LFOs*) generate digital control signals that can be used to change Note Pitch, Note Velocity and Controller values to create seemingly complex variations in your music.



The above diagram might look pretty complex but it's quite simple really. Starting at the left-hand side, the *LFO* generates a control signal – one of the 6 waveforms available on the front panel – and the *Master Depth control* acts as a kind of *volume control*, limiting the amount of *LFO* signal that gets through.

The same signal is then sent internally to the *Note Pitch, Note Velocity* and *Controller Streams*, all of which have their own individual *depth* controls.

Now, select a sequence, something simple with all 16 steps active and normal, ie. not *skipped,* because we want to keep this exercise as simple as possible. Make sure that this is the only sequence enabled at the time and make sure that you select the same sequence in the *Sequence Select* strip.



Let's begin by selecting an *LFO Waveform.* Choose the *SawUp* waveform to start off with. This will create slowly rising pitches and velocities that will drop down to their original values when the *LFO* resets back to the start. In this way the changes imparted by the *LFO* should be fairly easy to hear.

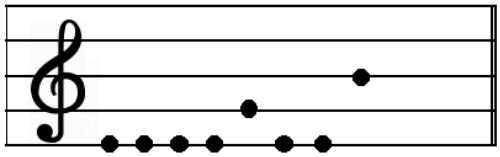
Now press the *Depth* push button in the *LFO Panel* and, using the *Data Wheel,* turn the *Master Depth* up to *75%.* Next, press the *Sync* push button. The LED above the push button will switch on to indicate that the frequency of the *LFO* is now slaved to the length of the sequence.

Now we're ready to hear something. Press *Play* in the *Transport Strip* and listen to the playback. It should be exactly as it was before, because even though we've turned up the *Master Depth* control in the *LFO*, we're not yet sending any of the *LFO's* signal to the individual streams.

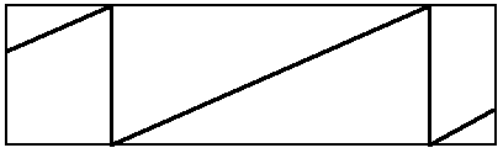
Go to the *Sequence Play Menu* and scroll through the pages until you come to the *Note Modulation* page.



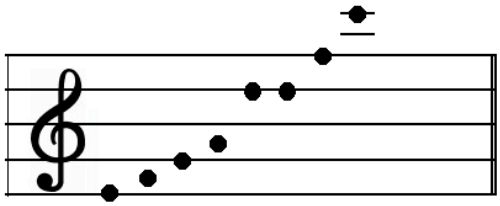
Using the *Data Wheel* slowly increase the *Pmod* parameter. You should begin to hear the pitch of the notes within the sequence change. Do the same with the *Vmod* parameter. If your synthesiser is playing a velocity-sensitive patch then you should start to hear the notes getting louder as the step counter moves towards the end of the sequence. When the *LFO* resets and the *sawtooth* waveform begins at 0 again, the note velocity will drop to its normal level.



Original sequence



LFO Output



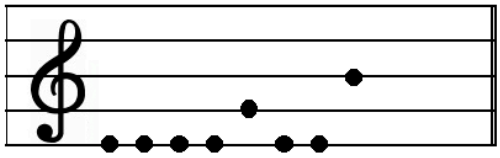
Modified sequence…

So, in a simple 8 note sequence, we should hear something similar to the above.

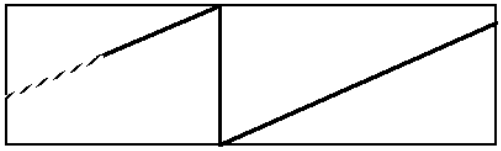
Next, we'll look at the *Time Offset* and *Depth Offset* parameters.The ***Time Offset*** *(TOFF)* field offsets the start of the oscillator cycle from 0 and is measured as a percentage of the *oscillator wavelength*. The ***Depth Offset*** *(DOFF)* field offsets the output level of the oscillator from the minimum possible value of 0.

That sounds a bit technical so let's actually hear what effect these two parameters have on our sequence.

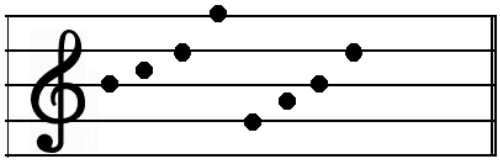
When we vary the *Time Offset* parameter, we're simple varying the point at which the *LFO* starts. Press the *TOFF* push button and change the parameter so that it reads 50%. As the diagram shows, varying *TOFF* can have a dramatic effect on the sequence and this can be a very powerful tool if you're attempting to create new sequences.



Original sequence



LFO Output



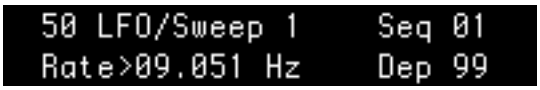
Modified sequence…

Now, decrease the *Time Offset* parameter so that we're back to our original sequence then increase the *Depth Offset* parameter. We're now adding a fixed amount of modulation to each *note pitch* and *velocity* so we should hear the pitch of the notes rising and the notes themselves getting louder.

As we've been using the *Time* and *Depth Offset* parameters, you may have noticed that the resulting sequences have been less than musical. This is because *ZEIT* adds the modulation data and picks the note nearest to the result, which might not be a note in your chosen key. The remedy to this problem is to set *Force-to-Scale* to on. If you haven't already done so, press the *Force-to-Scale* push button in the *Pitch/MIDI* panel and then you should find that no matter what you do with *TOFF* and *DOFF,* the pitches will fall within the current key.

In this example, we chose an *LFO Waveform* so that the changes to the note pitches and note velocities were quite obvious. We also locked the frequency of the *LFO* to the length of the sequence so that we wouldn't have to get involved with complex frequency settings. Suppose, instead, that we wanted to do something more subtle.

Press the *Sync* button once so that the LED goes off and the *LFO* is no longer synchronised to the length of the sequence. In fact, the frequency of the *LFO* is now fixed and won't change when we vary the playback tempo. Next, select the *Tri waveform* (short for *Triangle*) and press *Play*. The abrupt change generated by the *SawUp* waveform is replaced by more gentle, sweeping changes.



Press the *Rate* push button. The *Rate* parameter uses the *precision parameter editor* to change values, which some users find a little intimidating. To change the value, press the *Select* push button to move between the digits and the *Data Wheel* to increase and decrease the values. When you're finished, press the *Enter* push button to move to the next field.

When setting the *Rate* parameter, low frequencies mean long cycle times. A *Rate* of 0.100 Hz is equivalent to a cycle time of 10 seconds whilst 0.050 Hz is equivalent to a cycle time of 20 seconds. The minimum cycle time you can set is currently 0.010 Hz, which is equivalent to 100 seconds or nearly 2 minutes. Consequently, you can impart some very long, subtle changes to your sequences.

In the next tutorial, we'll look at combining what we've learned above with the *Controller Editors* to create more complex sequences.

Tutorial 3: Using the Controller Editors

In this tutorial, we'll show you how to get the best out of the two MIDI Continuous *Controllers*. We'll take you through setting up a series of controller values, setting the controller type and modulating the values using the LFOs.

This tutorial also follows on from the second tutorial and so we recommend that you work through *Tutorial 2* before attempting this exercise.

If you've never used MIDI Continuous Controllers then the best way to think of them is as a means of adding expression to your music, beyond simply playing the notes either louder or softer as you would with the note velocity controls.

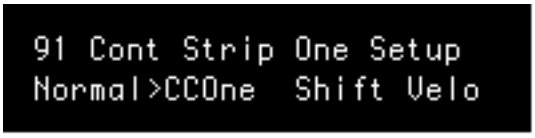
MIDI Continuous Controller messages have three components: a status byte that describes the MIDI Channel for this message, which is followed by the controller type and then the controller value itself.

There are two ways of inputting MIDI Continuous Controller information into *ZEIT*:

1) The Continuous Controller knobs
2) MIDI System Exclusive messages

In this tutorial, we'll cover just the first method.

Firstly, go to the *System* menu and scroll through the pages until you come to the *Controller One Setup* page. Change the setup so that *Controller Strip One* updates *CCOne* in *Normal* mode. Do the same with *Controller Strip Two* so that it updates *CCTwo* in *Normal* mode.



Exit the *System* menu and try changing the knobs on the *Controller One* strip. *ZEIT* will recognise the change and drop you into the *Controller Editor*.



Before we can do anything useful with this *controller stream* we need to make sure that the data stream is *enabled* - c*ontroller data streams* can be switched off to reduce congestion on the MIDI Out port.

Press the *Controller One* edit button to exit temporarily from this editor and then press this button a second time so that you're at the top level of the menu. (You'll find that this is a very handy shortcut for moving quickly between menus). Scroll through the pages until you come to the *Controller One Modulation* page.



Check that the third field, *Act*, is set to *on*.

Next, we'll set the *Controller Type*. There are 127 MIDI Continuous Controllers available within the MIDI standard, many of which are reserved for special purposes. Consult Appendix 2 at the end of this manual for a full list of the defined types. Although these definitions are *reserved*, you need to check with the manual that came with your instrument to find out how the manufacturer implemented the MIDI standard.

However, there are a number of *fixed* Continuous Controllers. One such *fixed controller* is *Controller 7* which, on the vast majority of MIDI systems, is used to update the master volume for a synthesiser patch. Another controller that is commonly used is *controller 74,* which can adjust the *Cut Off Frequency* of a synthesiser's filter.

Press the *Enter* button so that the cursor returns to the *Home* position and then scroll backwards until you come to the *Controller Type* page.



The default *Controller Type* for *Controller One* is *MIDI Volume (Coarse).* Press the *Enter* push button once and use the *Data Wheel* to scroll through the available controllers until you come to *Controller Type 74: Brightness*. *'Brightness'* is the term used in the MIDI Specification and this generally refers to the filter cut-off frequency. Now – and this is very important – to actually set the controller type, you must first press the *Select* push button. This isn't obvious and you need to remember it for future reference. Why is this implemented in such an unfriendly manner?

Suppose that *ZEIT* is playing a sequence as you're scrolling through the list of controller types. If the controller type changed every time you turned the *Data Wheel* then you could send a controller message that might suddenly update the portamento time, the pan position, volume level or the modulation amount for a synthesiser and that could potentially destroy a performance or recording session.

So, remember to press the *Select* push button when you want to set the *Controller Type*.

Now press *Play* and listen to the sounds made by your synthesiser. If everything is set up correctly and your synthesiser correctly responds to *controller 74 messages,* then you should hear the overall brightness of the synthesiser's sound vary as the sequence plays. Try changing a couple of the controller knobs and see what happens. If this doesn't work then we recommend that you have a look at the synthesiser's MIDI implementation to see how the instrument has been configured.

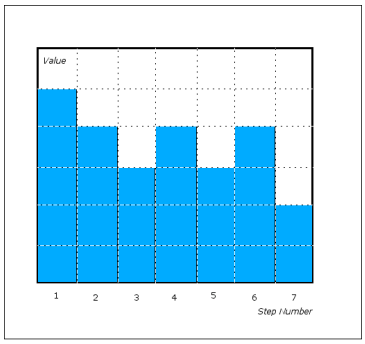
Just as with the *Note Editor* you can set up patterns of *active* and *skipped steps*.



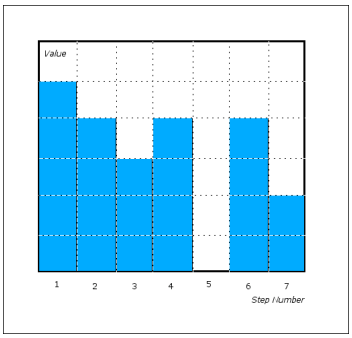
Try pressing a couple of the *active step* push buttons and listen to their effect. When a step is *muted*, no MIDI Controller message is sent and so the *controller value* stays the same as it was in the previous step.

When a step is *skipped* then the sequencer ignores the step and the *controller value* stays the same. You can use this technique to greatly reduce the amount of data in the MIDI stream. It also makes it very easy to add some really exciting and interesting rhythmic variations to your sequences.

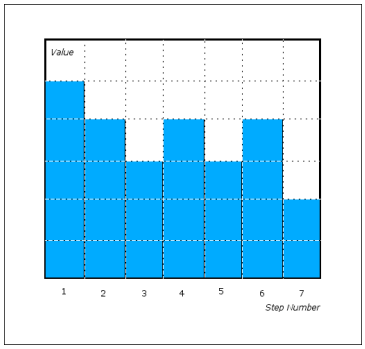
Experimentation is the key to success here, as is a flexible and well-specified synthesiser.



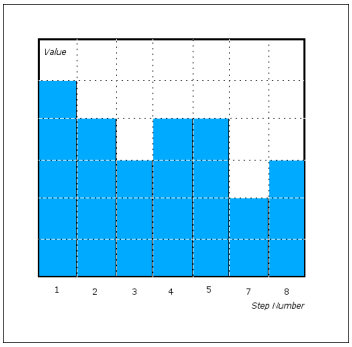
Before: All steps *Active*



After: step 5 *muted*



Before: All steps *Normal*



After: Step 5 skipped

Controller Two is identical to *Controller One* in every respect. For the next step in the tutorial, repeat all of the above with *Controller Two* except, for this section, select another *MIDI Controller Type*. For really interesting results, try changing the type to *5, Portamento Time* and then experiment with the results. You should find that you can create all manner of different playing techniques just by changing the *portamento time* in this way. Equally, you might like to try setting the *Controller Type* to number 1, *Modulation Wheel*. This will create some very intense, dramatic changes in most synthesisers.

In the next stage we're going to look at changing the *controller values* with the *LFO*. One of my favourite techniques is to use the MIDI Controllers in combination with the *LFO* and *Sweep Generators* to dynamically vary the filter cutoff frequency on a typical analogue synthesizer.

Most modern synthesisers make extensive use of MIDI Continuous Controllers and appendix 2 lists all of the common Continuous Controller settings such as Portamento Time, master volume and modulation wheel amounts.

If we look further down the list, we find controller 74, which can be used to set the *frequency cutoff* value for the filter. Next to it, we find controller 71, which is commonly used for changing the filter's *resonance*. This arrangement is fairly standard though some instruments do vary the combination so you should check with your synthesiser's manual if you don't get the expected results.

As before, we change the *Controller Type* for *Controller One* so that it updates continuous *controller 74*, labeled as *Brightness*. We'll change *Controller Two* so that it updates *continuous controller 71 aka Timbre.* If we now press *Play,* we should be able to hear the filter's cutoff frequency and resonance changing dynamically.

To use the *LFO*, press the *Controller One* menu button twice so that you're at the top level of the menu and then scroll down until you reach the *Controller Modulation* page.



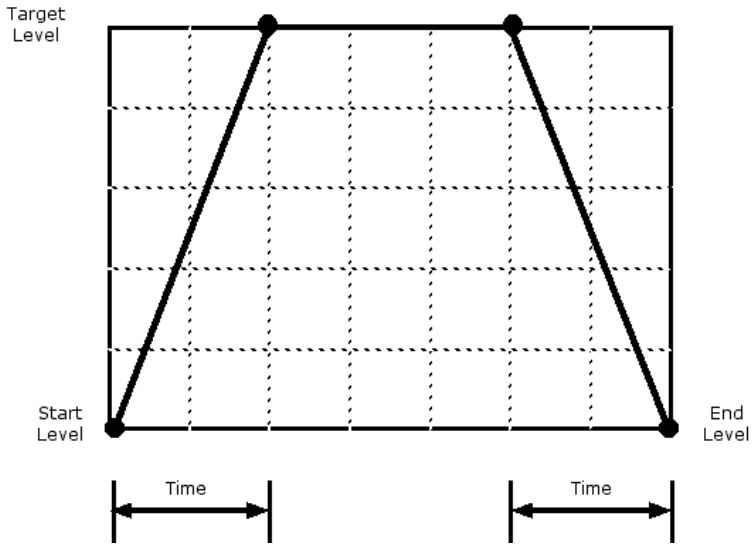
Increase the *Modulation Depth* parameter so that it's around 99% and then on the *LFO* strip select the *Random* waveform. This will create very dramatic variations in the filter's sound.

As we said above, you should be able to apply this technique to any *MIDI Controller Type* to create sequences that, though rhythmically related, never repeat.

In the final part of this tutorial, we're going to talk about using the *Sweep Generator* in conjunction with the *Controllers*.

The *Sweep Generator* controls are available in *Shift* mode on the *LFO Strip* and, in this example, we're going to create a slowly rising volume envelope. As in the previous examples, return to the *Controller Type* menu and choose *Continuous Controller Type 7*, which is the controller for MIDI Volume. Don't forget to press the *Select* push button to confirm the change.

Enter *Shift* mode by pressing the *Shift* key on the front panel and then press the *Start Level* push button. Set the *Start Level* to *0.* Press the *Target Level* push button and set it to 99% then press the *End Level* push button and set that to *0* as well.



Stay within the *LFO/Sweep Generator Menu* and move the cursor to the home position. Scroll through the menu pages until you come to the second Sweep Generator Modulation Routing page, as shown below. Make sure that the *Sweep Generator* modulation is switched *On* for both *CC01* and *CC02*.



Now go to the *Controller One* strip and turn all of the knobs to their minimum setting. Press *Play* and you should hear nothing. This is because the *Sweep Generator* is at its starting value, which we set to *0*.

Move the cursor to the *Home* position and scroll back up the menu pages until you come to the *Sweep Generator Enable/Disable* page.



The *State* parameter will show that the *Sweep Generator* is currently stopped. Press the *Enter* push button once and then use the *Data Wheel* to change the *State* to *Running*. The *Sweep Generator* will then begin running through the *Start Phase:*



The *Level* field will show the current value of the *Sweep Generator's Output*. When it reaches its maximum level, the *Sweep Generator* will stop and remain at the *Target Level*.

To begin the *Release* phase of the *Sweep Generator* press the *Select* button. The *Level* field will show a slowly decreasing value and will stop when the *Sweep Generator's* output reaches the *End Level.* At the same time, you should hear the volume of your MIDI sequence gradually reducing.

These features are complex and experimentation is the best way to work out how the sequencer works.

Tutorial 4: Morphing

Morphing is a technique that takes two sequences, works out the differences between them and, during playback, subtly changes the first sequence so that it sounds like the second sequence.

Sequence morphing can be used to create long evolving passages that do not repeat though the end result is always predictable and reproducible. If *force-to-scale* is switched on then the notes generated will always be in the currently loaded scale.

How Does Morphing work?

Let's choose a couple of simple examples. To begin, we need to specify a source sequence and a destination sequence. The source sequence is always the currently selected sequence whilst the destination sequence must reside in the battery-backed memory. Let's say that the source sequence is only four notes long and consists of the notes C, D, E and G. We'll restrict the notes in the destination sequence to four notes as well: D, C, A and D.

When you start the *morpher*, *ZEIT* works out the difference between the note pitches and the note velocities in the source sequence and the notes pitches and note velocities in the destination sequence (The *Controller values* are not changed).

Let's consider the first note in each sequence. In the source sequence this is the note C. In the destination sequence, this is the note D. The difference between the two notes is two semitones. The destination note, D, is higher in pitch than the source note, C and so the pitch change is going up the scale. We call this a *positive* change.

Now look at the second note in each sequence. In the source sequence this is the note D. In the destination sequence this is the note C. The difference between the two notes is again two semitones but, in this instance, the destination note is below the source note and so the direction of change is *negative*.

If we repeat this process for each note in the sequence then we can construct a table of values:

Source	C3	D3	E3	G3
Destination	D3	C3	A3	D4
Difference	2	2	5	7
Direction	+	-	+	+

To begin the process, the source sequence plays through once. At the end of this *pass*, *ZEIT* calculates a new value for the note pitch in each step. The modified sequence plays

through once and *ZEIT* again updates the note pitch for each step. This process repeats until the morphed source sequence is identical to the destination sequence.

So, what would the above sequence sound like and how many times would the sequences play through until the source and the destination sequence sound the same?

The time taken before the source sequence sounds like the destination sequence is based on the largest difference in pitch between the source and destination sequence. In the above example, the largest difference is 7 semitones and so the time taken for the source sequence to match the destination sequence would be (seven + one) passes, ie. 8 passes.

Let's assume that *force-to-scale* is not enabled.

Pass	Step No.			
	1	2	3	4
1 – Source Sequence	C	D	E	G
	+1	-1	+1	+1
2	C#	C#	F	G#
	+1	-1	+1	+1
3	D	C	F#	A
	0	0	+1	+1
4	D	C	G	A#
	0	0	+1	+1
5	D	C	G#	B
	0	0	+1	+1
6	D	C	A	C
	0	0	0	+1
7	D	C	A	C#
	0	0	0	+1
8 - Destination Sequence	D	C	A	D

So, what do we hear? On the first pass through, *ZEIT* would play the source sequence, C D E G. On the second pass through, *ZEIT* adds one semitone to the first note taking the C to a C#. The second note moves downwards in pitch from D to C#. The third note moves up in pitch from E to F and the fourth note also moves up in pitch from G to G#. So, on the second pass through we would hear the *morphed* sequence C# C# F G#. And so on, until the source sequence sounds like the destination sequence.

Ok, let's try using the same source and destination sequences except on this occasion, we'll switch on *force-to-scale*. *Force-to-scale* takes the incoming note and matches it against the current scale. If the note is in that scale then the note is not affected by *force-to-scale*. However, if the note is not in the current scale then the *force-to-scale* processor looks for the nearest note in the scale **below** the incoming note. For example, if you passed a C to the *force-to-scale* unit and the current scale was C major then the note would be unaffected. If instead, you passed a C# to the *force-to-scale* unit then it would output the nearest note below C# which is C.

Pass	Step No.							
	1		2		3		4	
1 – Source Sequence	**C**		**D**		**E**		**G**	
	+1		-1		+1		+1	
2	C#	**C**	C#	**C**	F	**F**	G#	**G**
	+1		-1		+1		+1	
3	D	**D**	C	**C**	F#	**F**	A	**A**
	0		0		+1		+1	
4	D	**D**	C	**C**	G	**G**	A#	**A**
	0		0		+1		+1	
5	D	**D**	C	**C**	G#	**G**	B	**B**
	0		0		+1		+1	
6	D	**D**	C	**C**	A	**A**	C	**C**
	0		0		0		+1	
7	D	**D**	C	**C**	A	**A**	C#	**C**
	0		0		0		+1	
8 - Destination Sequence	D	**D**	C	**C**	A	**A**	D	**D**

What's happening here? As before, on the first pass through, the source sequence plays C D E and G. On the second pass, the *morpher* adds one semitone to the first note, taking the C to a C#. However, the note C# isn't in the scale of C major and so this is *forced* to the new pitch of C, so there's no apparent change in pitch. However, with the second note in the sequence, the D is pitched downwards to C# and, as we've already seen, this isn't in the key of C major. Consequently, the D note is also *forced* to the note C. In the third note, the E is pitched upwards to F and, since F is in the scale of C major, then the *force-to-scale* function has no effect. Finally, the G is pitched upwards to G# and, as before, the *force-to-scale* function will re-pitch this note at G.

So, on the first pass we'll hear the source sequence C D E and G. On the second pass, we'll hear C C F and G (as shown in bold font in the table above).

Let's look at what happens on the third pass through. The first note, which is C#, is pitched upwards by a semitone to the note D. Here, the *force-to-scale* function has no effect. The second note, which was C#, is pitched downwards by one semitone to C and, again, the *force-to-scale* function has no effect. The third note, F, is pitched upwards to F# but modified by the *force-to-scale* function back to the original F. The fourth note, which was G#, is pitched upwards and becomes A. Hence, on the third pass, our *morphed* sequence plays D C F A.

Try setting this up for yourself using the *note editor* to set the various parameters.

Note Velocities

Are the *note velocities* affected by the *morphing* function? Yes, they are.

Let's stick with the current example where the source sequence is C D E G and the destination sequence is D C A D but, here, we'll give the notes a velocity component. The velocity components are shown in parentheses.

Source	C3 (50)	D3 (60)	E3 (70)	G3 (80)
Destination	D3 (50	C3 (74)	A3 (98)	D4 (38)
Velocity difference	0	+14	+28	-42
Direction	0	+	+	-

In the first note, there's no difference between the velocity values. In the second note, the source note D is played with a velocity of 60 and the destination is played with a velocity of 74. So, the difference between the two notes is +14. In the third note, the source velocity is 70 and the destination velocity is 98 and so there is a difference of +28. Finally, in the fourth note, the source note is played with a velocity of 80 and the destination with a velocity of 38 and so there is a difference of –42.

How are the note velocities *morphed*? Well, we saw in the previous example where *force-to-scale* was enabled that the total number of passes required to *morph* the source sequence into the destination sequence was 7. However, to *morph* the velocity components, we require one less pass and so we divide the velocity difference by the number of passes minus 1. So, in this example, the number of passes required for the velocity is 6.

And so the table above now looks like this:

Source	C3 (50)	D3 (60)	E3 (70)	G3 (80)
Destination	D3 (50)	C3 (74)	A3 (98)	D4 (38)
Velocity difference	0	+14	+28	-42
Direction	0	+	+	-
Velocity Increment	**0**	12/6 = **2**	24/6 = **4**	42/6=**7**

Now, what would this sound like?

	1	2	3	4
1	C(50)	D(60)	E(70)	G (80)
2	D(50)	C(62)	F(74)	A(73)
3	D(50)	C(64)	F(78)	A(66)
4	D(50)	C(66)	G(82)	B(59)
5	D(50)	C(68)	G(86)	C(52)
6	D(50)	C(70)	A(90)	C(45)
7	D(50)	C(72)	A(94)	D(38)

As the *morph* progresses then we'll hear the loudness of first step remain exactly the same whereas the second and third steps will gradually increase in volume and the fourth step will slowly decrease in volume.

Flipping

Suppose now that the source sequence and the destination sequence have different clock rates, different directions and different lengths. What effect does the *morpher* have on these parameters?

Let's look at a longer sequence.

Step Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Source	C	C	C	D	E	F	C	D
Destination	E	E	D	B	A	A	F	F
Difference	+4	+4	+2	+9	+5	+4	+5	+1

The maximum difference in pitch occurs at step four, where the source pitch is D and the destination pitch is B. The difference is 9 semitones, which means that the *morph* function will take (9 + 1) passes to complete.

We'll assume, for simplicity, that the source and destination sequences have the same directional settings. Here, both sequences are moving in the *forwards* direction.

	1	2	3	4	5	6	7	8
1 Source	C	C	C	D	E	F	C	D
2	C#	C#	C#	D#	F	F#	D#	D#
3	D	D	D	E	F#	G	D	E
4	D#	D#	D#	F	G	G#	D#	F
5	E	E	E	F#	G#	A	E	F
6	E	E	E	G	A	A	F	F
7	E	E	E	G#	A	A	F	F
8	E	E	E	A	A	A	F	F
9	E	E	E	A#	A	A	F	F
10 Destination	E	E	E	B	A	A	F	F

Now, let's change the direction of the *destination* sequence so that it's now moving backwards.

The most logical place to *flip* the direction and/or clock rate is at the mid-point in the *morph* and the midpoint is calculated by dividing the total number of passes by 2. If the total number of passes was 10 then the mid-point would fall at the end of pass 5, which is where the *flip* would take place.

When the *morph* begins, playback will be in the *forwards* direction. After pass number 5, the playback direction will change to that of the destination sequence i.e. backwards. What effect will this *flip* have on the above table? Let's see.

	1	2	3	4	5	6	7	8
1 Source	C	C	C	D	E	F	C	D
2	C#	C#	C#	D#	F	F#	D#	D#
3	D	D	D	E	F#	G	D	E
4	D#	D#	D#	F	G	G#	D#	F
5 (*flip!*)	E	E	E	F#	G#	A	E	F
6	E	E	E	G	F	F	A	A
7	E	E	E	G#	F	F	A	A
8	E	E	E	A	F	F	A	A
9	E	E	E	A#	F	F	A	A
10 Destination	E	E	E	B	F	F	A	A

Suppose the source sequence is clocked at a rate of 2x and the destination sequence is clocked at a rate of 4x. The *morph* would begin at the source clock rate of 2x. It would remain at this clock rate until the end of pass 5, upon which it would change to the destination clock rate of 4x.

Active and Muted Steps

Let's use the previous example, this time, with active and muted steps:

Step Number	1	2	3	4	5	6	7	8
Source	C	C	C	D	E	F	C	D
Active	Y	Y	Y	Y	Y	Y	Y	N
Destination	E	E	D	B	A	A	F	F
Active	Y	Y	N	N	Y	Y	N	N
Difference	+4	+4	+2	+9	+5	+4	+5	+1
Flip pass	3	3	2	5	3	3	3	2

First, let's see what the sequence would sound like if all of the steps were active. For simplicity, *force-to-scale* is not enabled and the shaded cells indicate that the step has reached its target pitch.

	1	2	3	4	5	6	7	8
1 (Source)	C	C	C	D	E	F	C	D
2	C#	C#	C#	D#	F	F#	D#	D#
3	D	D	D	E	F#	G	D	E
4	D#	D#	D#	F	G	G#	D#	F
5	E	E	E	F#	G#	A	E	F
6	E	E	E	G	A	A	F	F
7	E	E	E	G#	A	A	F	F
8	E	E	E	A	A	A	F	F
9	E	E	E	A#	A	A	F	F
10	E	E	E	B	A	A	F	F

As before, the maximum difference in pitch occurs at step four, where the source pitch is D and the destination pitch is B and the difference, in semitones, is 9.

However, step four in the destination sequence is muted and so *ZEIT* must decide when to mute step four. *ZEIT* works out the difference in pitch between the source step and the destination step and divides this value by two, adding one if the total number of passes is an odd number. So, step four will play in pass one, two, three and four but will be muted on the fifth pass.

What would playback sound like? In the following table, the step number runs horizontally and the pass number runs vertically. As before, the greyed-out boxes indicate steps where the morphed pitch matches the destination pitch. The blacked-out boxes indicate muted steps.

	1	2	3	4	5	6	7	8
1	**C**	**C**	**C**	**D**	**E**	**F**	**C**	
2	C#	C#	C#	D#	F	F#	D#	
3	D	D	D	E	F#	G	D	
4	D#	D#		F	G	G#		
5	E	E		F#	G#	A		
6	E	E			A	A		
7	E	E			A	A		
8	E	E			A	A		
9	E	E			A	A		
10	**E**	**E**			**A**	**A**		

So, the *morphed* source sequence will be the same as the destination sequence after only 6 passes and so the entire *morph* operation will be considerably shorter.

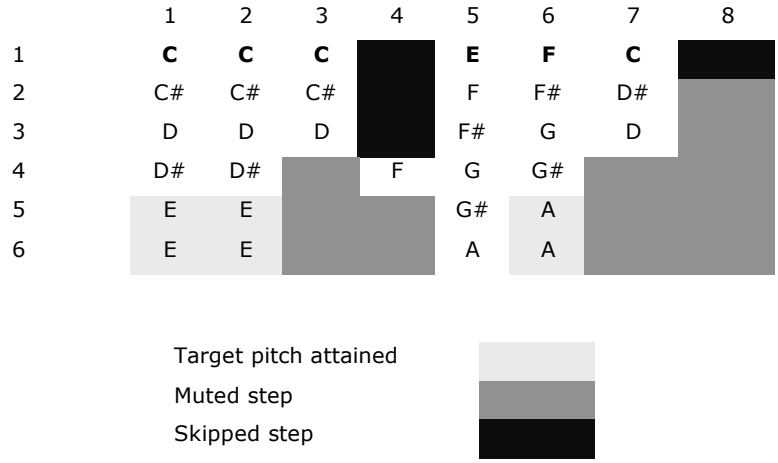
Skipped Steps

Skipped steps require careful consideration. Let's look at the previous example again, this time replacing some of the muted steps with *skipped steps*.

Step Number	1	2	3	4	5	6	7	8
Source	C	C	C	D	E	F	C	D
Active	Y	Y	Y	Y	Y	Y	Y	N
Skipped	N	N	N	**Y**	N	N	N	**Y**
Destination	E	E	D	B	A	A	F	F
Active	Y	Y	N	N	Y	Y	N	N

Skipped	N	N	**Y**	N	N	N	**Y**	N
Difference	+4	+4	+2	+9	+5	+4	+5	+1
Flip pass	3	3	2	5	3	3	3	2

And the sequence would play as:



Looping and Morph Modes

In normal circumstances, a *morph* would simply begin with the source sequence and proceed until the source sequences sounds the same as the destination sequence. Once the *morph* has completed, the destination sequence will play until the sequencer itself is stopped. However, you can make a *morph* restart simply by enabling *loop mode*.

When looping is on, the source sequence is morphed until it matches the destination sequence. When it matches the destination sequence, the process is reversed so that the destination sequence becomes the source sequence. The morphing process then begins to work back towards what was the original source sequence. We've found that this method creates some very smooth, evolving passages.

Morph Dwell is the number of times the destination sequence plays once the morph process has completed. If this parameter is set to, say, three then the source sequence will morph until it matches the destination sequence. The destination sequence will then play three times before restarting the morph back towards the original source sequence.

Hints and Tips

Morphing is a great way of finding new sequences that are closely related to the sequences you're currently working on. We've used it many, many times to create a large number of subtle variations based around an original theme.

To do this, pick your source and destination sequences, start the ***Morpher*** with the ***data wheel*** and hit the ***Play*** button. When you hear a sequence variation that you like, simply turn the ***data wheel*** one step anti-clockwise to stop the ***morphing process*** at that point. The sequence will continue to play as normal and will not be over-written by the next progression until you start the ***morpher*** running again. This leaves you with time to manipulate and save the sequence as required.

Restarting the ***Morpher*** will start the process off again with your original sequences.

Some Morphing questions

What happens to the Controller Data?

The governing factor in the morphing process is always the note values. The controller values are generated and processed without modification. However, you can still change them from the front panel controls, the low frequency oscillator, the sweep generator etc.

Can I have more than one morph running at one time?

Unfortunately, this is not currently possible. Morphing is very hungry on processor resources and, at present, *ZEIT* is not able to generate more than one morph. This may change in future software revisions.

Morphing only seems to work with the Forward, Reverse and Pendulum directions?

Yes, this is true. As stated, morphing is a complicated process. Adding randomised and algorithmic directions to the process takes up too many CPU resources. Again, this may be addressed in future software revisions.

This page intentionally left blank.

Tutorial 5: Using the Quick Select Facility

One of our basic design goals for the *ZEIT* sequencer was to give the performer the ability quickly and accurately to change sequences, patches and tempo *on the fly* in a way that the old-style analogue sequencers did not. The *Quick Select* facility provides these facilities and, in this tutorial, we'll look at setting up tempo changes, active/mute changes, loading sequences and *force-to-scale* records to that you can load them without the need to go scrolling through menus.

Quick Select records are configured as 8 banks of 8 records, giving a total of 64 locations. The bank select function occupies the first 8 buttons on the *menu strip* and the memory select function occupies the second 8 buttons.

At only 3 pages long, the *Quick Select Menu* is one of *ZEIT's* shortest menus. The first page let's you name a *Quick Select Bank*, the second changes the record type and the third sets up the data. That's all there is to it!

However, setting up the *Quick Select* facility needs some time and patience and so, to get the best out of your *ZEIT* sequencer, you should read this section through and then you'll find that using an analogue-style step sequencer in a live situation has never been easier.



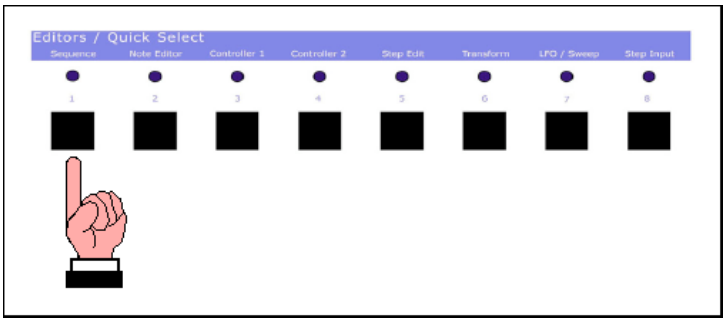
Before we begin, we need to make sure that pressing one of the *Quick Select* buttons pulls up a *Quick Select* record. By default, the *Quick Select Facility* is switched off because it gets in the way of editing parameters via the normal *Edit Menus* and the *Quick Select Facility* is usually used in *Performance* situations where editing parameters within menus is not always a priority.

Make sure you're in *Normal* mode and press the *Menu Button* labelled *System*. If the *System Setup Menu* appears then scroll down to page 4 and set the *QuickSelect* parameter to *on*. If you pressed the *System Setup Menu* push button and a *Quick Select record* appeared then you're ready to go.

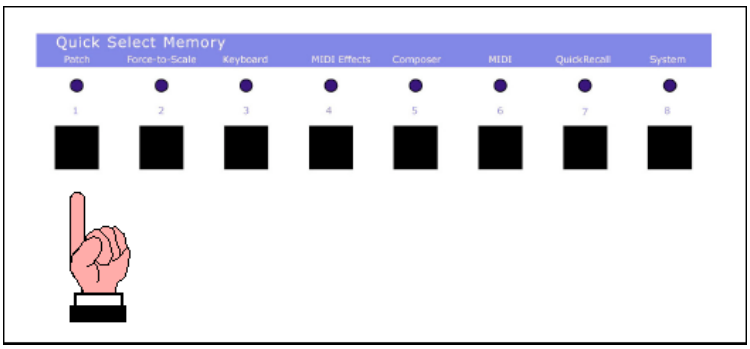
For the purposes of this tutorial, we'll still need access to the *QuickSelect Setup Menu* and we can do this via *Shift Mode*. When the front panel is in *Shift Mode*, pressing one of the *Quick Select push buttons* always pulls up the relevant *Edit Menu.*

Now we're ready to start setting up some *Quick Select* records.

First, press the first push button in the *Quick Select Menu Strip.* This will load in the first *Quick Select Record Bank.*



Next, press the first button in the *Quick Select Memory Bank.* This will load in the first record, ready for editing.



Let's begin by setting up a *Quick Select Bank* for a particular song. Firstly, we'll give the *Quick Select bank* a name - let's call it something imaginative like *DemoSongBank*.

Press the *Shift* push button so that the front panel is in *Shift Mode* and then press the *Quick Select Menu.*



Next, go to page 1, press the *Enter* push button and *rename* the *Quick Select Bank* in the usual manner. When you're finished, press the *Enter* button a second time to return the cursor to the *Home* position.

Suppose now that our demo song features a couple of *tempo changes* between the *introduction, verse* and *chorus*. Rather than using the *Data Wheel* to vary the *playback tempo* of the sequencer, which is a very slow process, we can use the *Quick Select* facility to change the tempo with just the touch of a button.

We'll designate *button 1* for the *introduction*, *button 2* for the *verse* and *button 3* for the *chorus.*

Scroll down to the second page in the *Quick Select* menu and use the *Enter* push button to move the cursor to the *Type* field. Now use the *Data Wheel* to set the record type to that of *Tempo*.



Move the cursor back to the *Home* position and scroll down to page 3 in the *QuickSelect* menu.



As before, use the *Enter* push button to move the cursor to the *parameter* field and use the *Data Wheel* to set the *playback tempo* for the *Introduction* to the song, in this case 100 BPM. That's the *Introduction* taken care of. Now we'll move on to the *Verse* section of our song.

Move the cursor back to the *Home* position and scroll back to page 2. Now move the cursor down to the *But* field and use the *Data Wheel* to select *Button Number 2.*



Again, if it isn't already set, change the *Button Type* to *Tempo*.

Scroll forwards to page 3 again and change the tempo for the *verse* to 120 BPM.



Repeat the above for the *chorus* section, changing the tempo to 130BPM.

Now we're ready to play. Press the *Shift* mode button so that the front panel returns to *Normal* mode and then jump back to the *Patch/Tempo* page. Next, press the first *Quick Select Memory Button.* The playback tempo in the *Patch/Tempo* page will now change to the *tempo* we set above i.e. 100 BPM.

Press *Play* in the *Transport* strip and listen to the playback. At an appropriate point, press *Button 2* in the *QuickSelect Memory bank* and the tempo will jump to the playback speed for the *verse i.e. 120BPM.* Finally, press *button 3* in the *Quick Select Memory strip* and the playback speed will jump to the tempo we set for the *chorus* section.

Key changes

Now, suppose that our demo song contains a key change, perhaps a major key to a minor key. We can use the *Quick Select* facility to load in the *scale records* without having to go through the *force-to-scale* menu.

Press the *Shift* button again and then call up the *Quick Select Menu*. Scroll back to page 2, change the button number to 4 and set the *type* field to *Scale*.



Now, go to page 3 and select one of the available scales. For the current song, choose *Major*.



Repeat the above for button 5 only, this time, set a different scale, say *Minor*.



As before, return to *Normal mode* and then switch back to the *Patch/Tempo* page. Press button 1 to set the *playback tempo* for the *Introduction* and then button 4 to set the current scale to '*Major*'. Now, press *Play* in the *Playback Strip.*

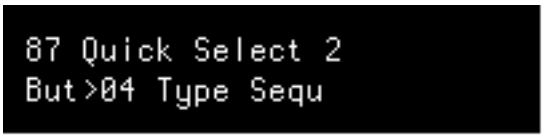
Play the song through a couple of times and listen to the effect of the various buttons.

> ***Hint:*** *Suppose you want to change the tempo and the scale at the same time? The order and timing of the operation is pretty critical but we've found that performing the scale change immediately after the last note in the sequence has played and then pressing the tempo change button on the next downbeat gets the best results. It takes practice but it does work!*

Sequences and Patches

Next, we're going to look at setting up a couple of *Quick Select Records* so that you can load in either sequences or patch records on the fly.

Call up the *Quick Select Menu* and select button *6*. Go to page 2 and change the type to *Sequ* (for *sequence)* and then go to page 3 so that you can select a suitable sequence.



Do the same with button *7*, this time selecting a different sequence.



On playback, you can change the sequence as often as you like without having to go through the *sequence menu*.

> ***Hint:*** *You may notice that it's a little difficult to press several Quick Select Memories at once. It is possible, with a little practice but there is an easier way. When the Quick Select facility is enabled, you can send a quick succession of MIDI program changes from an external sequencer and they will have exactly the same effect.*

You can also set up a bank of *Quick Select* buttons to change *Patch* in exactly the same way. However, *Patch Records* do take time to load and so requesting a number of *Patch Changes* in quick succession may produce some unexpected results.

Active and Muted Steps

Finally, we'll look at setting up a collection of *active* and *muted* steps so that you can quickly change patterns during playback.

First, call up *Quick Select Bank 2* and press button 1 in the *Quick Select Memory strip* so that we're dealing with a new bank of records. Use page 1 to rename this bank to something useful, say *PatternChanges*.

Next, go to page 2 and change the record type to *Act*, which is short for *Active*. Now go to page 3, move the cursor down to the *Act* field and then press the *Select* push button. The pattern of *active* and *muted* steps on the currently selected sequence will be copied into the data field for this *Quick Select record*.



Now, use the front panel switches to set up another pattern of *active* and *muted* steps. When you're happy with the new pattern, return to the *Quick Select Menu*, go to page 2 and make sure that the record type is the same as record 1, i.e. *Act*. Finally, jump to page 3, go to the *Act* field and press *Select.* The modified pattern of *active* and *muted* steps will be copied to the data field for this *Quick Select* record.



As before, press *Play* and try changing the pattern of *active* and *muted* steps using the *Quick Select Push Buttons.* You should find that this is an excellent way of introducing rhythmic changes into a sequence quickly and without pressing a lot of buttons.

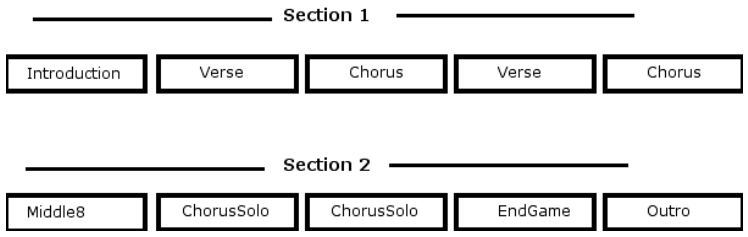
You can apply the same technique when setting up patterns for the *skipped* and *normal steps*, the *effected* steps for the MIDI Effects unit and the *enabled* and *disabled sequences*.

The *Quick Select Facility* is a fast, simple and effective method of enhancing your musical performances. We strongly recommend that you get acquainted with these facilities because the range of possibilities is simply mind-blowing.

Tutorial 6: Using the Composer

In this tutorial, we're going to look at using the ***Composer*** to create a simple song from a collection of sequences. We'll use the Note Editor to enter the sequences and the Composer to chain the sequences together to form a song.

The song in question is a track entitled *Dust Devils* and a version of this track is available for download from the Infection Music web site for those who want to hear what it sounds like in advance. It's a fairly simple, dancy, trancey sort of thing, which may or may not be your cup of tea. However, don't let that put you off. The techniques used to put together *Dust Devils* can be applied to any other genre of music just as easily.



Let's start by looking at the overall structure of *Dust Devils*. The whole piece will consist of 27 **measures**, each of 16 steps. Each measure will be made up of one **part** (phrase), which may be the same as or different from the previous part – some parts may repeat more than once. Each part is made up of the 4 **tracks** corresponding to a particular voice part, eg. main sequence, kick drum, snare drum and bass line.

The first half of the piece (measures 1-16) contains an introduction, a verse and a chorus. The second half (measures 17-27) begins with the middle-8, a slightly reworked version of the chorus for the main solo and then finally, the end game/outro parts where we finish everything off.

ZEIT's Composer is limited to just 4 tracks and so we need to do a bit of forward planning so that everything fits together neatly. It often helps to sketch the structure of a composition on a bit of paper before you get in too deep. I like to look for short cuts –parts that repeat, parts that are silent etc.

Anyway, let's start at the beginning.

The Introduction

The first sequence in Part 1 is just a single 16-note sequence. Enter the notes below using either the Note Editor or Step-Input and save the sequence as *DustSequence*.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
C	C	F#	F#	**C**	C	F#	F#	**C**	C	F#	F#	**C**	C	F#	B

This is the only sequence that we want to have playing in Part 1 so the other 3 tracks must be silent. The simplest way to create a period of silence is to create an empty sequence – one where all of the steps are *muted*. Simply edit a normal, 16-step sequence like *DustSequence* but *mute* every step. The start and end steps are still 1 and 16 respectively and so the result is 16 steps of silence!

> ***Hint:*** *You can save a great deal of time and effort by creating a small number of blank sequences, which can be reserved for use with the Composer. For instance, Blank16 is 16 steps long, all of which are muted. Similarly Blank8 consists of 8 muted steps. However, remember to use the* ***Sequence Lock*** *facility in the Sequence Play Menu, which will prevent these from being accidentally over-written.*

The kick drum sequence, like the blank sequence mentioned above, is another really useful template for speeding up the compositional process. As with the sequence part, either dial it in with the pitch knobs or use the Step-Input facility. Save this sequence as *BasicKickDrum*.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
bd	HH	**bd**	HH	**bd**	HH	**bd**	HH	**bd**	HH	**bd**	HH	**bd**	HH	**bd**	HH

It often helps to create a couple of versions of this sequence, each with a slightly different feel. For instance, you can create just a basic kick drum with the same note on every step or, you can create a sequence with the kick drum on every odd step and a hi-hat on the even step for a dance/trance feel. The possibilities are endless. It's always a good idea to give these sequences a useful name like *kick16steps* and *kickhihat16* to minimise confusion later on.

The third track will be reserved for the snare drum. Here, we'll simply create a series of notes that begin at step 8 and gradually rise in velocity to produce the beloved machine-gun effect.

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
								sn	sn	sn	sn	sn	sn	sn	sn

Here, the darker shading indicates a greater note velocity. Save this sequence with the name *BasicSnareDrum.*

> ***Hint:*** *For a more realistic feel, try using the Step-input menu when creating drum parts. Subtle differences in note velocities feel far more fluid and less mechanical. Unless, of course, you're after a very machine-like feel!*

The final sequence we will need for the Introduction is the bass synthesiser part. This is very simple and consists of just 4 steps. Here's the table of notes:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
C	C	F#	F#	**C**	C	F#	F#	**C**	C	F#	F#	**C**	C	F#	B

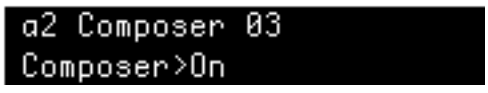
Save this sequence as *DustDevilBass1*.

We now have all four sequences for the *Introduction* to our specimen song *Dust Devils* but, before we can proceed, we have to make sure that the *Composer* is ready for use.

Press the *Composer* button on the front panel (Quick Select Strip) and use Page 1 to select a free song. Rename the song using Page 2. In this instance, the name is *'DustDevils'*.

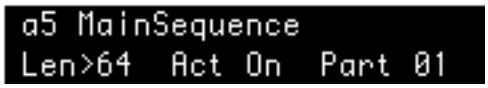


Next, scroll down to page 3 and make sure that the **Composer** field is switched *on*.



The Composer is now ready for use. However, before we start assigning sequences to tracks, we can make life easier if we assign a useful name to each one.

Scroll down the menu to page 4, the *Select Track* page, and select *Track 1*. Next, go to Page 5 and rename *Track 1* to something useful such as *MainSequence*. Scroll down to page 6 and enable *Track 1, Main Sequence,* by setting the **Act** field to *on.*



Repeat this operation for the remaining tracks, 2, 3 and 4. Call them *DustKickDrum, DustSnare* and *DustBassLine* respectively.

Finally, go back to page 4 (*Track Select*) and reselect *Track 1, MainSequence*.

All this may seem a bit tiresome, and indeed it is. You can still use the Composer without going through any of the tedious renaming bits but giving everything a proper name at this point makes it easier to follow the tutorial!

Assigning Sequences to Parts

Now we'll start to arrange the sequences.

In the Introduction, *DustSequence* plays through twice and the remaining three tracks are silent for the first 2 measures. In this example remember, a measure is 16-steps.

	Measure 1 Part 1	Measure 2 Part 2
1.	DustSequence	DustSequence
2.	Blank 16	Blank 16
3.	Blank 16	Blank 16
4.	Blank 16	Blank 16

Go to page 8, the *Select Part/Seq* page and move the cursor to the *Part* field. Use the *Data Wheel* to select *Part Number 01*. Press the *Enter* push button once to move the cursor to the next field and use the *Data Wheel* to select the sequence *DustSequence*.



Press the Enter button twice so that the cursor again moves around to the **Part** field and use the Data Wheel to increase the *Part Number* to 2. As before, press the Enter button to move the cursor around to the next field and use the Data Wheel to again select the sequence *DustSequence*.

We now have 2 parts, each of which will play the sequence *DustSequence*. The next step is to make each of the remaining tracks play the appropriate blank sequences! So, go back to page 4, the *Track Select* page, and select *Track*2, the *DustKickDrum* track. Repeat the process described in the previous paragraph so that this track plays 2 measures of silence by selecting the sequence *Blank16* for each measure.



Finally, when you've set all of the tracks in both parts to the appropriate sequences, press the *Reset* button in the Transport Strip and then press *Play*. If everything went according to plan, you should hear the sequence *DustSequence* on its own.

That's the first half of the *Introduction* taken care of. Now, we'll tackle the second half of and add a bit of detail to the composition.

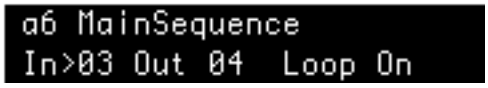
Here, we introduce the remaining voices, the kick drum, snare drum and bass line.

	Measure 3 Part 3	Measure 4 Part 4
1.	DustSequence	DustSequence
2.	BasicKickDrum	BasicKickDrum
3.	Blank 16	BasicSnareDrum
4.	DustDevilBass1	DustDevilBass1

The kick drum and bass sequence both begin at the third measure. The snare drum sequence starts at measure 4. Begin by setting up track 1 to play the *DustSequence* through on measures 3 and 4. Then, move to track 2 and set up Parts 3 and 4 so that they play the sequence *BasicKickDrum*. Track 3 still plays a silence during Part 3 but the *BasicSnareDrum* will begin at Part 4. Finally, the bass sequence, *DustDevilBass1*, plays both Parts 3 and 4.

As before, press *Reset* in the Playback strip and then press *Play*. You should hear the whole of the *Introduction* playing together.

We said earlier that these parts were a little basic and that it might be a good idea to create some variations based around these basic ideas. Well, this is the place to try them out. We can do this by putting each track into *Loop* mode so that they'll repeat until the sequencer is stopped. Also, you won't need to keep hitting the *Reset* button to go back to the start.



Go to page 7, the *Track Status* page*,* and set the *In* and *Out* Parts to 3 and 4 respectively, and then set the *Loop* parameter to *on* for each track. As before, press *Reset* followed by *Play*. This time, the Composer will play through from the first part in each sequence but will begin looping around Parts 3 and 4.

Now you can use page 6, the *Part/Sequence Selection* page, to change the assigned sequence for each part.



Here, we're substituting the *BasicSnareDrum* sequence with a variation on the original sequence just to add a bit of colour and variety.

The First Verse

The first verse of *DustDevils* is not significantly different from the Introduction. However, we'll use a new parameter, the *Repeat Count* so that we don't have to enter each part, one by one.

The first verse of *DustDevils* is 4 measures in length. So, as before, go to track 1, the *MainSequence* track and set up Part 5 so that it plays the sequence *DustSequence*. Then, go to page 9, the *Edit Part* page and set the *Repeat Count* to 4.



Repeat this process for each of the 4 tracks in the first verse so that you finish up with a structure that appears as follows:

	Measure 5 Part 5	Measure 6 Part 5	Measure 7 Part 5	Measure 8 Part 5
1.	DustSequence	DustSequence	DustSequence	DustSequence
2.	BasicKickDrum	BasicKickDrum	BasicKickDrum	BasicKickDrum
3.	SnareDrum2	SnareDrum2	SnareDrum2	SnareDrum2
4.	DustDevilBass1	DustDevilBass1	DustDevilBass1	DustDevilBass1

The measures outlined by a dotted line are repeats. Notice that for the verse, a variation of the original snare sequence is used, *SnareDrum2,* just to add a bit of variation and to help distinguish it from the Introduction.

You should also notice that using the *Repeat* parameter is a more efficient way of creating songs. You just have to remember which parts play and for how long. If you want to make each part different then you can simply stick with the first method we described.

The First Chorus

We can use the same approach when creating the chorus*,* which is not wildly different from the verse. In fact, the only real difference is that the bass sequence and main sequence transpose up the scale by 2 semitones. We can accomplish this very easily by using the *Transpose* parameter, which is also on the *Edit Part* page.

To help differentiate the chorus from the verse and to help propel the piece forwards by injecting a bit of energy we can make a couple of changes to the original percussive parts. Firstly, use the Note Editor to adjust all of the kick and snare sequences manually so that they are a little bit louder. Then use the *Randomise* facility in the Transform menu to vary the bass and main sequences. Make sure that force-to-scale is enabled.

	Measure 9 Part 6	Measure 10 Part 6	Measure 11 Part 6	Measure 12 Part 6
1.	DustSequence2	DustSequence2	DustSequence2	DustSequence2
2.	KickDrumHiHats	KickDrumHiHats	KickDrumHiHats	KickDrumHiHats
3.	SnareDrumLoud	SnareDrumLoud	SnareDrumLoud	SnareDrumLoud
4.	DustDevilBassX	DustDevilBassX	DustDevilBassX	DustDevilBassX

So, this is the structure of the Chorus section. Again, the *Repeat* facility in the Edit Part page is used so that we don't have to set up each part individually. Notice that although we're at Measure 9 in the piece, we're still editing Part 6.

To transpose the sequence parts, we again use the Edit Part page.



It's important to remember two things here. Firstly, don't transpose the percussive parts *KickDrumHiHats* and *SnareDrumLoud* because they probably won't sound right and, secondly, switch each track out of *Loop* mode using the *Track Loop* page otherwise *ZEIT* will loop around the wrong parts and you won't hear the chorus at all. Alternatively, change the *In* and *Out* parts so that *ZEIT* loops around the Chorus instead.

Our song, *DustDevils*, now consists of two introductory sections, a verse and a chorus. We can use exactly the same techniques discussed above to create a second verse and a second chorus. As before, try playing around with each of the sequences to make them slightly different though not enough to make them unrecognisable, just enough to retain the feel and content of the piece without throwing away the essential elements.

The Middle 8 and Solo Sections

The Middle 8 is deliberately simple and consists of little more than the basic kick and snare sequences used in the Introduction. We don't really need to consider these much, except to say that in the real piece, the break was used for two purposes: firstly, to improvise some ambient chords around the main theme and, secondly, as a queue to remember to set up the external synths ready for the Solo.

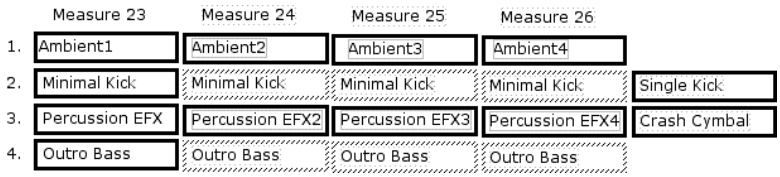
	Measure 17 Part 7	Measure 18 Part 8	Measure 19 Part 9	Measure 20 Part 9	Measure 21 Part 9	Measure 22 Part 9
1.	Blank16	DustSequence2	DustSequence2	DustSequence2	DustSequence2	DustSequence2
2.	BasicKickDrum	BasicKickDrum	KickDrumHiHats	KickDrumHiHats	KickDrumHiHats	KickDrumHiHats
3.	BasicSnareDrum	BasicSnareDrum	SnareDrumLoud	SnareDrumLoud	SnareDrumLoud	SnareDrumLoud
4.	Blank16	DustDevilBassX	DustDevilBassX	DustDevilBassX	DustDevilBassX	DustDevilBassX

The Solo is, essentially, a modified version of the Chorus although the drums were beefed up a bit and the Note Editor was used to add some bite to the first note in every bar. Again, these were all saved as separate sequences.

The Middle 8 is made up of two individual parts created in the same way as we created the Introduction. The Solo section is defined once, at Measure 19, and each part repeats 4 times.

And to the End...

The End section is the simplest section of all. Here, we create a basic kick drum playing once every bar instead of the usual *four-on-the-floor* pounding. We also create some new sequences to generate some additional percussive effects and apply these to the snare track. They're all individual sequences and so we don't use the *Repeat* function.



For the *Main Sequence* track we create some variations on another theme and chain them, one after the other, each transposed by an octave to create a dynamic, improvised feel.

Finally, to finish the whole track off with a bang, we create two new sequences, which consist of little more than a single kick drum and a single crash cymbal, a simple way of adding a full stop to the piece.

And that is *DustDevils*. To listen to the original version plus a couple of other variations we put together based on the original sequences, log on to the Infection Music Web site, click on *ZEIT Sequencer* and look for *DustDevils* in the Downloads section.

Hopefully, you should have seen that the Composer is a very powerful tool for quickly creating ideas and arrangements from very simple elements.

If you would like us to host your own version of *DustDevils* then please get in touch with us at the Infection Music Web site.

Tutorial 7: Using the Transform Menu

The *Transform Menu* is the place to go when you're stuck for ideas. We've already discussed one aspect of the *Transform Menu*, *Morphing,* in another tutorial and this chapter covers some of the other functions available to you.

Let's start with an easy example. Suppose you've been working with a sequence in the *Note Editor* and it sounds really good but the sequence starts in the wrong place, say step 5. It would sound much better if it started on step 1 so that it lines up with all of the other sequences. You could use the *Note Editor* to set the pitch and velocity for each step to the desired value but that would take time and might not work properly.

The *rotate* function in the *Transform Menu* can help remedy this situation. Let's look at a real sequence.

1	2	3	4	5	6	7	8
G7	**F7**	**F7**	**G#7**	**G7**	**D#7**	**C6**	**C8**
9	10	11	12	13	14	15	16
A#7	**G7**	**A#7**	**C8**	**G7**	**D#7**	**C7**	**D#7**

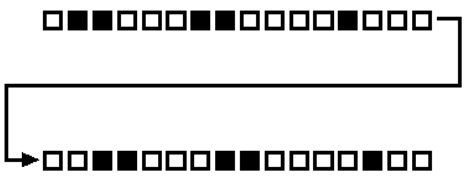
When playing this sequence, step 7 (C6) appears to be the best starting point for the sequence, so to move step 7 to step 1, we can use the *rotate* function to make the change for us.



In this case the sequence is 16 steps long, so when we enter the *Transform Menu* we can leave the *Start Step* and the *End Step* at the default values.

To perform the *rotate* operation, press the *Select* push button once. This will move all of the steps in the sequence one place to the right. In other words, step 2 becomes step 1, step 2 becomes step 3 and so on up to step 16, which takes on the value of step 15. What happens to step 16? Does it just fall off the end? No. The sequence wraps around to the start so that step 1 takes on the values of step 16. Our sequence now looks like this:

1	2	3	4	5	6	7	8
D#7	**G7**	**F7**	**F7**	**G#7**	**G7**	**D#7**	**C6**
9	10	11	12	13	14	15	16
C8	**A#7**	**G7**	**A#7**	**C8**	**G7**	**D#7**	**C7**



We can *undo* this operation if we press the *Shift* push button and press the *Select* push button again. The sequence will now sound the same as it did when we first entered this option because we've *rotated* the sequence back in the opposite direction.

Now, we originally wanted to move the start step of our sequence so that it began on step 1 whereas it is currently at step 7 (C6). To do this, we need to move the sequence to the left a further 6 times. If you repeat the operation above a further 6 times you should end up with a sequence that looks like this:

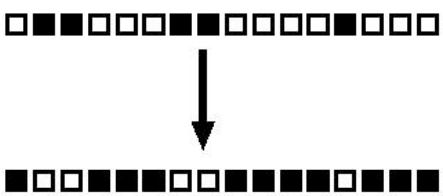
1	2	3	4	5	6	7	8
C6	**C8**	**A#7**	**G7**	**A#7**	**C8**	**G7**	**D#7**
9	10	11	12	13	14	15	16
C7	**D#7**	**G7**	**F7**	**F7**	**G#7**	**G7**	**D#7**

Before you proceed to the next step, save this sequence away. We'll need it later.

Inverting



Next, we'll look at the *invert* function. Choose a sequence with an interesting rhythmic feel - in other words, a sequence with a few steps muted and/or skipped. Select the *Xform Invert* page and press the *Select* push button once. Any step that was *active* is now *muted* and vice versa. Any step that was *normal* is now *skipped* and vice versa. Listen to the results. It's possible to create some very unusual rhythmic patterns with this technique.



To *undo* this operation, press the *Select* push button again.

1	2	3	4	5	6	7	8
C6	**C8**	**A#7**	**G7**	**A#7**	**C8**	**G7**	**D#7**
9	10	11	12	13	14	15	16
C7	**D#7**	**G7**	**F7**	**F7**	**G#7**	**G7**	**D#7**

Now, what effect would this have on our example sequence? All of the steps are active and all of them are normal. *Inverting* this sequence would cause all of the *active* steps to become *muted* and all of the *normal* steps to become *skipped* and so we'd end up with a sequence that did nothing!

So, let's do something more useful instead. Let's *mute* all of the even numbered steps whilst leaving the odd numbered steps *active.*

1	2	3	4	5	6	7	8
C6	**-**	**A#7**	**-**	**A#7**	**-**	**G7**	**-**
9	10	11	12	13	14	15	16
C7	**-**	**G7**	**-**	**F7**	**-**	**G7**	**-**

Return to the *invert* page within the *Transform* menu, start the sequencer running and press the *Select* push button once. All of the *muted* steps become *active* and all of the *active* steps are now *muted.* The sequence now sounds like this:

1	2	3	4	5	6	7	8
-	**C8**	**-**	**G7**	**-**	**C8**	**-**	**D#7**
9	10	11	12	13	14	15	16
-	**D#7**	**-**	**F7**	**-**	**G#7**	**-**	**D#7**

Press the *Select* push button again and the sequence reverts back to its former state. So, the *invert* function gives you a quick way of switching between two patterns very quickly indeed.

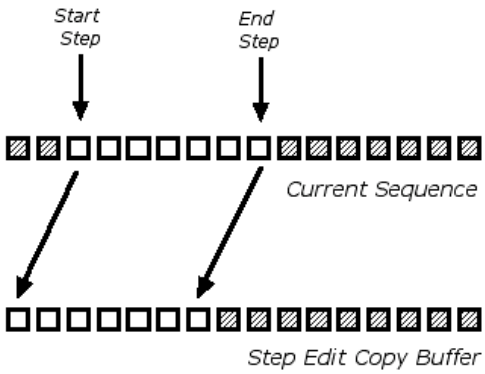
Merging



Merging two sequences is another way to create new patterns with interesting rhythmic and melodic properties. To use the *merge* facility, we first need to make sure that the *Step Edit Copy Buffer* contains some useful data. *Merge* won't work without it.

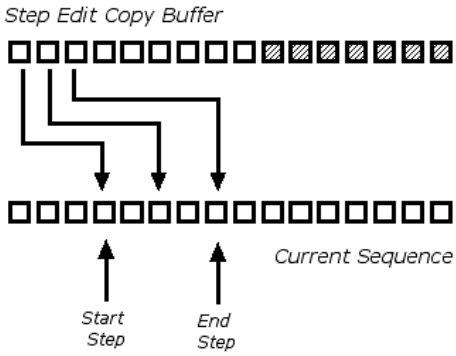


Go to the *Step Copy* page of the *Step Edit* menu and change the *start* and *end* steps so that they're different from the *default* values, say, 3 and 9. Press the *Select* push button once so that the steps between the *start* and *end* points are copied over to the *Step Edit Copy Buffer.* Step 1 in the *Step Edit Copy Buffer* now holds the contents of step 3 in the source sequence, step 2 holds the contents of step 4 and so on.



Next, load in a different sequence using the *sequence play* menu and then return to the *Xform Merge* page within the *Transform menu.* Set the *start step* and the *end step* to, say, 4 and 8 respectively then press the *Select* push button once. *ZEIT* will *merge* the two patterns together.

In the previous paragraph, we set the *start* and *end steps* to 4 and 8 respectively. Steps outside the *start* and *end steps* will not be affected by the *merge* operation so steps 1, 2, 3, 9, 10, 11, 12, 13, 14, 15 and 16 won't change.



What happens to steps 4, 5, 6, 7, and 8? Well, the *merge* operation will copy step 1 from the *Step Edit Copy Buffer* and paste it over the top of step 4 in the current sequence. Step 5 is ignored and so retains all of its values. Step 2 in the *Step Edit Copy Buffer* overwrites step 6 in the current sequence whilst step 7 is ignored. Finally, step 3 in the *Step Edit Copy Buffer* overwrites step 8 in the current sequence and the *merge* is complete. That's

sounds complicated but the end result is that the current sequence contains elements of the original sequence plus the current sequence.

Note Wrapping



Finally, let's look at *note wrapping*, a technique that can modify the sound of a sequence either very subtly or very dramatically.

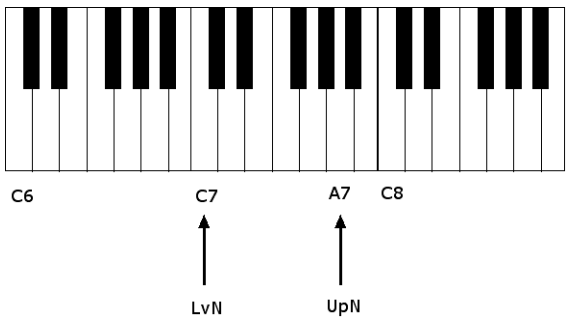
Before we begin, make sure that the *wrap function* is switched on. Move the cursor to the *Wrap* field and use the data wheel to set the status to *on*.

1	2	3	4	5	6	7	8
C6	**C8**	**A#7**	**G7**	**A#7**	**C8**	**G7**	**D#7**
9	10	11	12	13	14	15	16
C7	**D#7**	**G7**	**F7**	**F7**	**G#7**	**G7**	**D#7**

Let's return to the sequence we saved above.

The lowest note in the sequence is C6 and the highest note is C8. What would happen if we applied a *note wrap range* of C7 to A7?

Return to the *note wrap page* in the *Transform Menu* and move the cursor to the *LwN field.* Use either the data wheel or a MIDI keyboard to set the lower and upper thresholds for the *wrap function* to *C7* and *A7* respectively.



Step 1 has a pitch of C6, which is one octave below the *LwN* value of *C7*. The *Note Wrap* function works out the difference between the note pitch and the value of *LwN* and subtracts the difference from the *UpN* parameter. So, in this instance, the difference is 12 semitones and when this is subtracted from *UpN* yields a new pitch of *A6.*

Step 2 has a pitch of C8, which is 3 semitones above the *UpN* parameter. When this is added to the *LwN* parameter, a new pitch of D#7 is generated.

Step 3 has a pitch of C#7, which is only 1 semitone above the *LwN* value. If we subtract this difference from the *UpN* value then the new pitch is G#7.

So, the *Note Wrap* function gives us one more tool for generating new and exciting sequences and patterns in real time. It is worth spending some time and effort learning to use these tools. The usefulness or otherwise of these facilities may not, at first, be readily apparent. However, we've found that they're an excellent way to generate new ideas and new directions for your music, which might perhaps take you somewhere that you wouldn't have otherwise explored.