

Reaktor Tutorial



A Tutorial on Granular Synthesis

Part 01 version 0.1

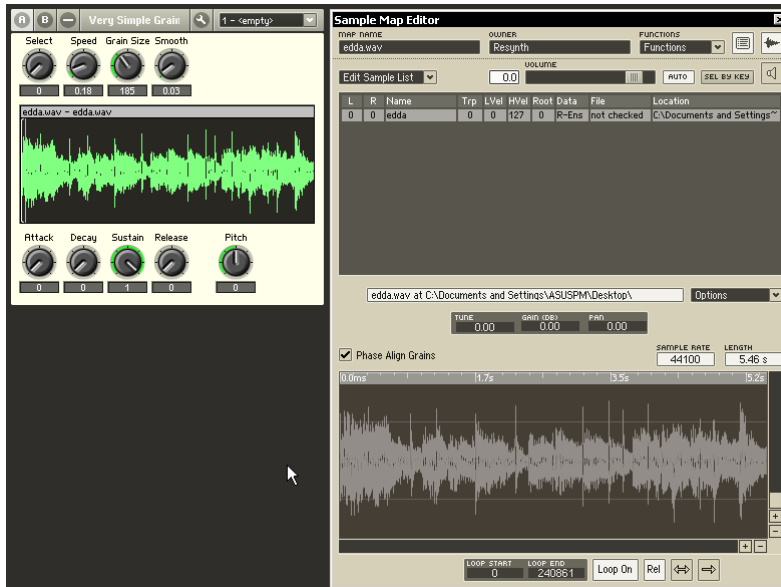
By Peter Dines

peterdines@gmail.com

<http://reaktortips.blogspot.com/>

Welcome to part 1 of the granular synthesis tutorial. I hope anyone interested has already downloaded the Very Simple Grainer instrument and had a good play with it, because we are going to be looking at ways to improve it. If you aren't already familiar with the original instrument, download it and do some exploring now; otherwise, you will be befuddled and nonplussed.

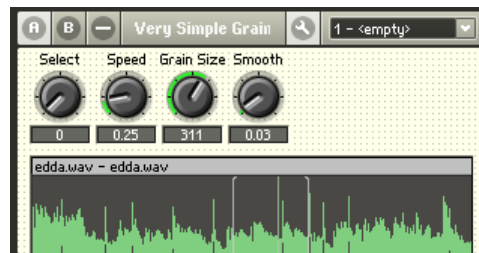
Open up Very Simple Grainer 01.ens. The first thing you will notice in this version is that the loop start and loop end knobs have gone missing.



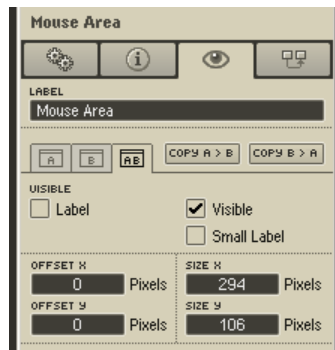
Also notice that the sample map property “loop on” is set to positive for this sample. That was true in the original grainer – I just wanted to point it out. If you don't have loop on, the loop start and end won't work. If you add more samples, you have to do the same for them.

So what's taking the place of the loop start and loop end knobs? Well, now we can click on the waveform display to set loop start and end points. You set the loop start by left clicking, and the loop end by right clicking. How does that work? With the help of a mouse area module.

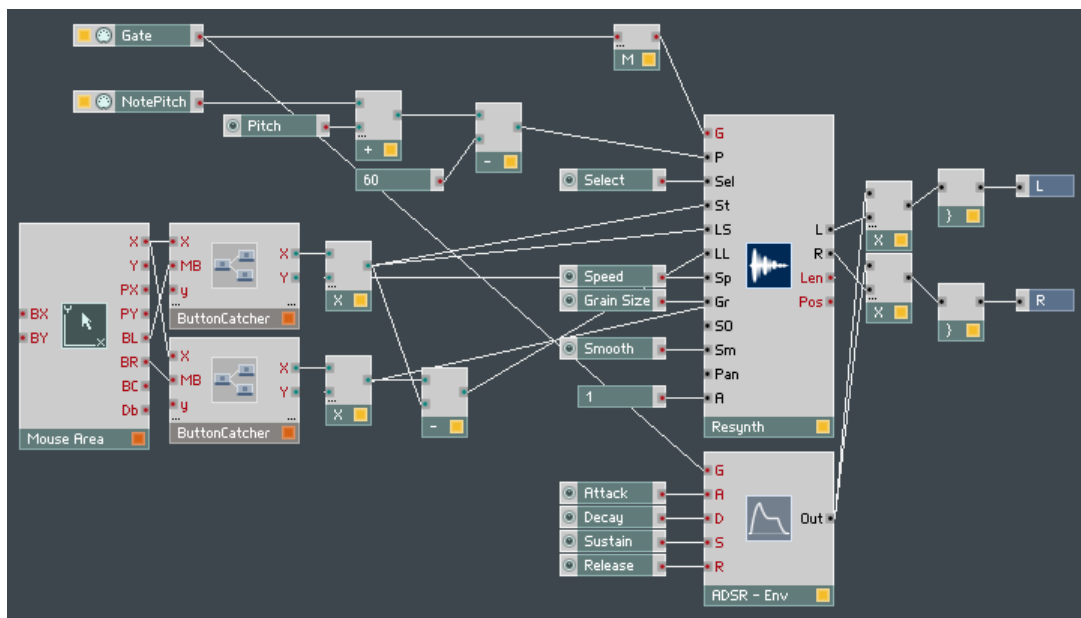
Click on the wrench icon on the instrument header to display the mouse area. It appears as a hazy translucent object over the waveform display when the instrument is unlocked, so you can see where it is. It becomes completely transparent when the GUI is locked.



In the mouse area display properties, its X and Y size is set to the size of the waveform display plus two pixels in each direction. That gives just the right amount of coverage.



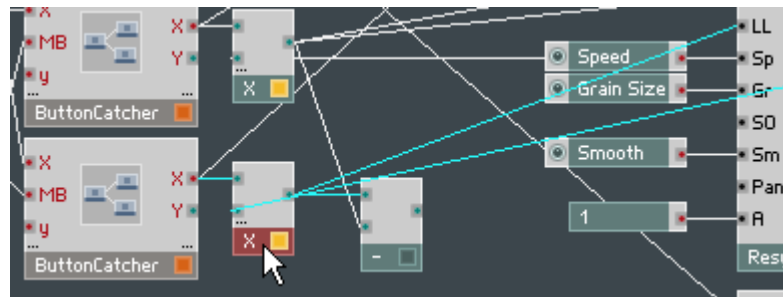
Now let's take a look at the instrument's inner structure:



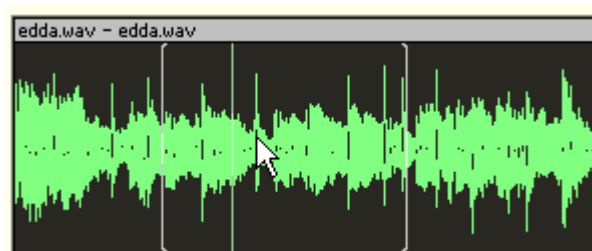
The X, BR (button right) and BL (button left) outputs on the Mouse Area are connected to two ButtonCatcher macros, which I will explain shortly. The X outputs on the ButtonCatcher macros are attached to two multiplier modules. So the default zero to one range from the Mouse Area is multiplying and therefore scaling... what?

It's multiplying the Len output of the Resynth module. The Len output gives the length of the current sample in milliseconds. So when you multiply the length of the sample in milliseconds by a range between zero and one, you get a percent value of the length. When the Mouse Area sends an X value of 0.5, that represents a position halfway through the sample.

So we have the left button controlling the loop start and the right button controlling loop length. Why is the loop start value being subtracted from loop length? Try bypassing it – connect the right button multiplier directly to the LL input on the Resynth module and see what happens.



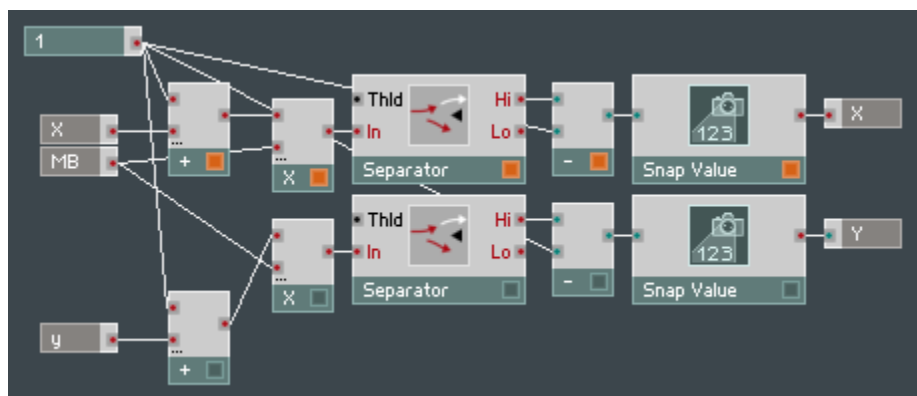
Without that offset, your right button clicks no longer correspond to loop end position:



Reconnect the subtractor module to the LL input of the Resynth (or ctrl-z to undo the connection) and we're good to go.

Now what about those ButtonCatcher macros? A problem with using the X output of the Mouse Area to control loop start and end is that your instrument has to know which parameter you are setting when the X value – the horizontal value – changes. Otherwise your loop start and end are set to the same value and you have a zero loop length.

There's more than one way to solve this problem. This is mine. Drill down into the top ButtonCatcher – the one that sets start and loop start, and here's what you will see:



Does it look complicated? Relax. We can immediately ignore the bottom half, because we aren't using the Y value of the Mouse Area, so it's disconnected. See? I've already doubled your productivity. :-)

The Separator modules in the middle of the chain are the key to keeping the X value from going through the macro when the corresponding mouse button is not depressed. The Thld (threshold) input determines the point at which events are split and sent to either the Hi or Lo outputs. With nothing connected to Thld, its default value is zero. So any event at the input with a value greater than zero goes to Hi, and any event with a value equal to or less than zero goes to Lo. You'll notice that Lo isn't connected to anything in this macro, so those events are discarded.

The MB input receives a value from the Mouse Area module, which sends a value of 1 when the button is depressed and zero when it's not. Multiplying the X value – the position of the mouse over the mouse area – by 1 or 0 determines whether or not the X position values will make it past the Separator module.

Why, you may well ask, is a value of 1 added to the X value before the Separator module and subtracted again immediately afterwards? Because we want to be able to set the loop start to zero. If a zero value came into the macro it would be rejected by the Separator and you'd never be able to set the loop start to zero – only to almost-zero. And that would be Simply Wrong.

Why is there a Snap Value module just before the output ports of the ButtonCatcher macro? Because the Mouse Area module does not remember its values and save them in a snapshot with the instrument. Putting Snap Value modules here lets the instrument remember your loop settings in each individual snapshot. This was not needed when the loop points were set with knobs, whose positions are stored with snapshots by default.

That's all we're going to cover in this installment. Stay tuned to <http://reaktortips.blogspot.com> for more upgrades (upgrains?) to the Very Simple Grainer.

Questions or comments? Post them on the blog or email me at peterdines@gmail.com