BEATCHOP

REALTIME SLICER AND REARRANGER

version 1.2.0

Operation Manual

QUADELECTRA AUDIOWORX

Re Rack Extension

➘ 1. Welcome To BeatChop

Thank you for purchasing *BeatChop Realtime Slicer & Rearranger* Rack Extension for Propellerhead Reason. We would like to welcome you to our first Creative Effect for the Reason Rack.

This new Rack Extension will enable you to treat audio like never before.

Behind BeatChop there's a whole new concept for sound processing, which blends the best abilities from Beat Repeaters, Slice Loop Players and Samplers in to one single effect.

We sincerely hope, you enjoy using BeatChop.

Please note: This manual covers the 1.2.0 version of BeatChop.

1.1. So What is BeatChop?

To put it in simple words: You can imagine BeatChop as a device similar to Dr.OctoRex but with the key difference, that it slices and maps the real-time input audio it receives to your MIDI keyboard!

Here's how it works: When your song plays, BeatChop continuously records the incoming audio from its inputs, to an internal buffer. This buffer has an adjustable size, that you can set to any musical time measurement you desire. In particular you set the size of the buffer in number (count) of steps and resolution (time units) such as 1/8, 1/16 etc, just like you'd do it with Reason's own Matrix.

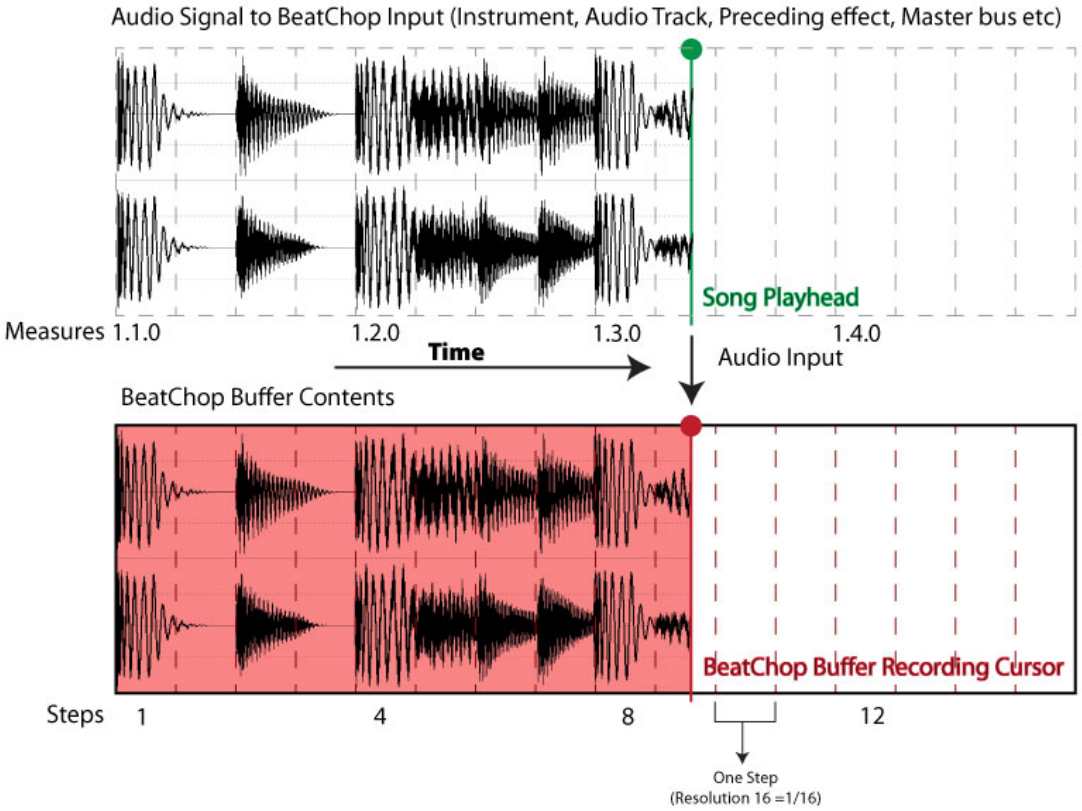


Fig.1.1-1: Audio Input is recorded to the BeatChop's internal buffer, in tight-sync with your song. So each bar's 1.3.2 segment will always be recorded to step #9 assuming you have a 16 steps x 1/16 buffer setup.

Once BeatChop has recorded audio to the internal buffer from start to end, it repeats this cycle replacing the old audio contents with new. BeatChop is aware of the current song position, and stores the audio data in tight sync with your arrangement. This means that if you wish to record one bar of audio using a 16 step buffer of 1/16ths, the fourth 1/16th of each bar will always be written to the 4th step of BeatChop's buffer.

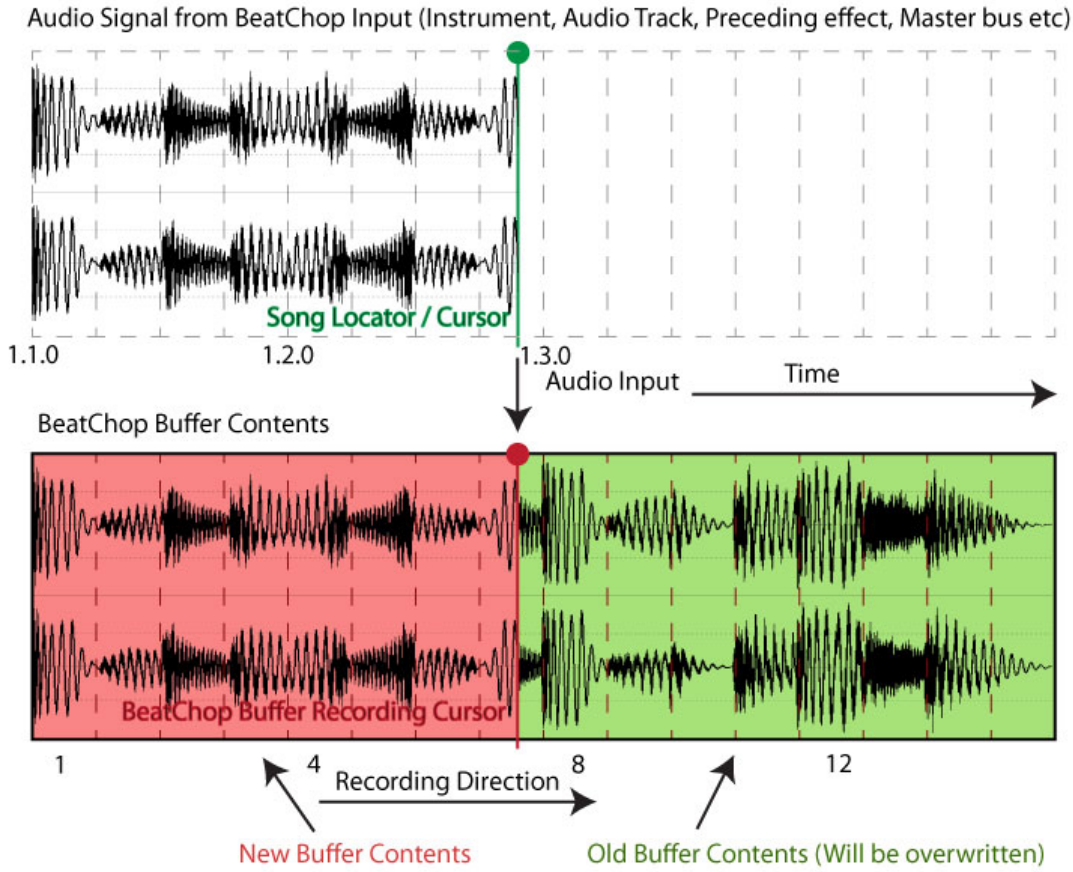


Fig.1.1-2: When the entire buffer is written from start to end, BeatChop repeats the Recording Cycle replacing old audio content with new...

As soon as the audio is recorded, it's automatically sliced according to the step count and the resolution you've selected. The produced slices are then available for playback via MIDI, and you get to decide if, how and when to trigger them, like you would do on a Dr. OctoRex.

You can shuffle, play in forward, reverse, or random order, or even arpeggiate these slices, but also you can loop them, make glitches, tape stops, back-spins etc. And all these using the real-time audio of your track.

No re-sampling, editing, slicing or layering necessary! BeatChop does that for you!

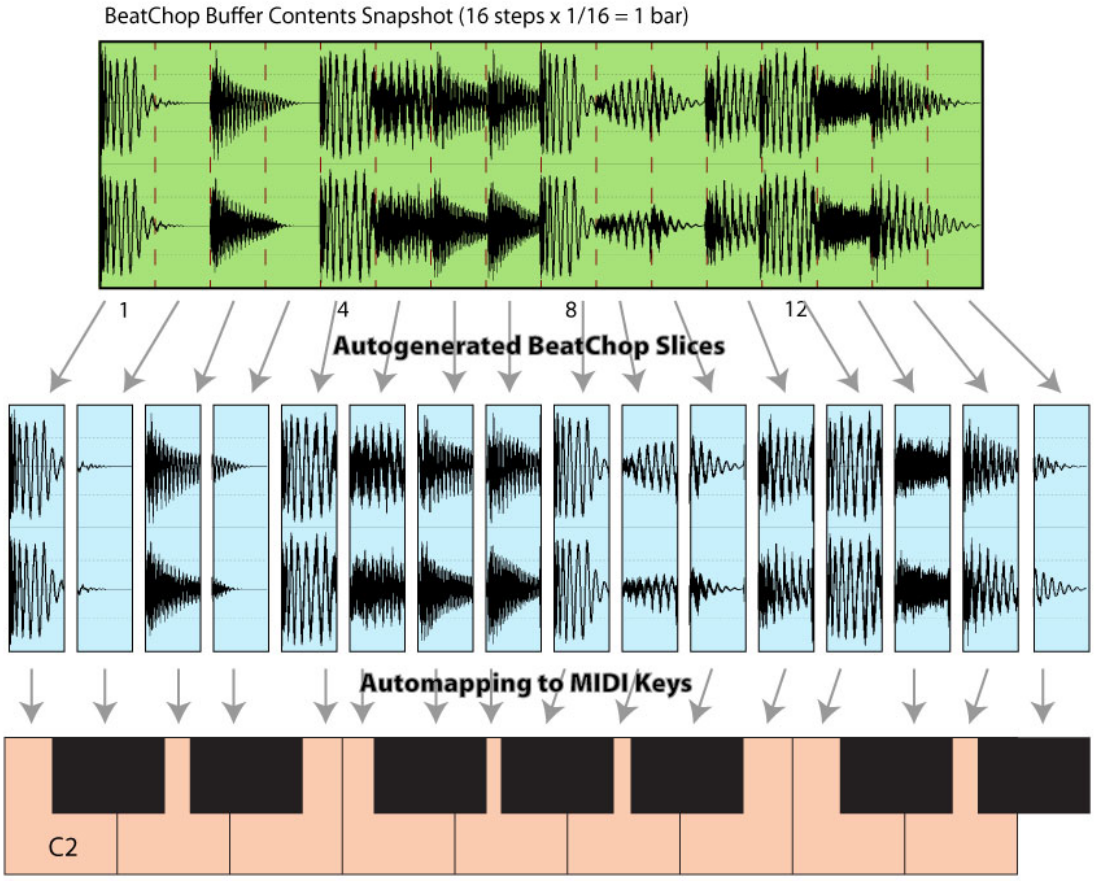


Fig.1.1-3: Contents of the buffer are sliced according to step count and resolution. Then they auto-mapped to the MIDI keyboard.

We wanted to make BeatChop a smart and versatile tool, a device that will inspire you to make music (and of course glitch it right after) :)

You are kindly advised to read this manual in order to understand the full potential that this device has, and how to get the most of it's capabilities.

➘ 2. Getting To Know BeatChop

In order to better understand what BeatChop can do for you, there is a need to take a closer look on how it all works. In chapter 2 we focus on some concepts behind the device, and its functionality.

2.1. The Buffer

As stated before BeatChop's internal buffer records the incoming audio in a loop, replacing old audio material with new. You have the ability to set the size of the buffer (how much time of audio is recorded) by changing the buffer's step count and the step resolution.

2.1.1. Recording To Buffer:

We've discussed before that BeatChop writes the incoming audio in tight-sync with your Reason Sequencer. This means that incoming audio can be recorded to the buffer, only during play time. If you stop the device, the buffer will stop to record also, and will retain the currently stored material, which you'd be able to use.

This happens because without the sequencer playing, BeatChop does not have a reference of synchronization to understand where inside the buffer should any new audio be written. On the other hand, when playing your song BeatChop synchronizes to the song position so it knows at which location in the buffer, incoming audio should be placed.

2.1.2. Roll Mode (Keeping The Buffer Contents)

BeatChop's buffer has a "Roll" function, very similar to that of "The Echo" delay unit. In BeatChop enabling "Roll" function will simply stop the buffer from recording new audio when your song plays, thus keeping any recorded material at that time, for further use.

Its important to understand that contents stored in the buffer, with or without using the "Roll" function will *not* be saved along with your song.

In other words you should *not* expect BeatChop to save the buffer audio, just as you wouldn't expect "The Echo" or any delay unit for that mater, to save the feedback audio from Rolling.

We'll discuss more on the Roll function later on this chapter.

2.1.3. Downsampling:

Although at most you can have up to 64 steps - 1/1t each, BeatChop's internal buffer is limited to 4 seconds. In sample count 4 seconds, is 4 times the sampling rate.

If the product of your step count and resolution exceeds 4 seconds, BeatChop will start downsampling new audio to the most appropriate speed,

in order to fit the entire recorded audio, to the maximum number of samples available to the buffer.

2.1.4. Tempo Change

BeatChop conforms to tempo changes by relocating the record and playback cursor to the appropriate position, according to song synchronization and sampling / writing speed.

Normally under these circumstances, successive changes to the tempo, would produce heavy artifacts to the audio for as long as they took place. To avoid this, BeatChop does not record audio while the song tempo is changing.

2.2. Slices

Although the terminology is "borrowed" from Propellerhead Recycle, BeatChop's slices have some key differences to those of a ReCycled Loop.

2.2.1. Polyphony

BeatChop slice playback is monophonic. This has to do with the nature of this effect as an "audio re-arranger" creative effect rather than an instrument.

2.2.2. Fixed Lengths

In contrast to ReCycle Loops that are pre-sliced according to transients, all BeatChop slices have the same length, as set by the "Resolution" parameter from the buffer. For that reason you should always pay special attention to quantization inconsistencies from live audio material.

2.2.3. Playback & Pitch

BeatChop can playback slices either in forward or reverse. There are also 2 different loop modes: Either directional (forward) or bi-directional (alternating).

BeatChop also provides 3 different controls for changing the pitch of the slices: The Pitch Bend Wheel, A Semitones parameter for transposing up or down, and a speed knob to alter the speed (and therefore the pitch) of the slices in a linear manner. Also there are 2 dedicated octaves for transposing up and down on the MIDI keyboard (see: APPENDIX I).

2.2.4. "Safe" Mode

BeatChop reads from and writes to its internal buffer in Real Time.

Under normal circumstances this behavior could cause some audio distortion due to overlapping between the read and written portions. To prevent this BeatChop "enters" in a kind of "Safe Mode" and will disallow to pitch the currently recorded slice higher.

This happens only when all of conditions below are met:

1. Slices are being played forward (i.e. not reverse).
2. Buffer is not set to "Roll"
3. The Reason arrangement is playing.

2.2.5. Slice Start & Slice Length

While Slice Start Offset is, as one would expect, an offset to the actual slice start, slice length denotes the distance between start offset and the end of the slice.

This means that a slice length of 50% to a slice with start offset 50%, will only play the 25% of the actual slice starting at the second half of the slice.

The same principle applies for quantized values. So 1/2 the length, of a slice that starts at 1/2 will play only a quarter of the slice starting at the second half.

In addition BeatChop has a set of four dedicated MIDI notes, ranging from G#0 to B0 that allow you to change the length setting on the fly in a set of predefined, relative to the slice length, values.

2.2.6. Envelope

BeatChop conforms amplitude for all slices to an ADR (Attack, Decay, Release) Envelope generator.

2.2.7. Auto-mapping

For performance convenience, BeatChop always utilizes *all* 64 available slice MIDI keys (C2 - D#7 / See APPENDIX I), regardless of the number of steps you set the buffer to. When the last slice is mapped to a MIDI key, BeatChop will restart mapping the next, to slice #1 and so on, until all 64 slice keys are mapped.

2.3. Performance Switches

Performance switches are a series of four functions that you can control either by their corresponding switches in the front panel, or by MIDI (see: APPENDIX I).

These are:

- **Roll Buffer:** Buffer Roll when enabled will cause the buffer to stop recording new audio. This allows you to reuse any audio that was recorded till you enabled, and for as long as you enable the Roll Switch.

- **Retrigger:** Enable slice retriggering. When this switch is enabled BeatChop will always start playing back each slice from the first sample. By disabling retrigger BeatChop will take in account any delay between the time the slice was supposed to be played (f.e. the start of a 1/16 if the buffer resolution is 16), and the actual time you pressed the MIDI key, and start the sample by offsetting to that difference.

- **Reverse:** Enabling this function will cause all slices to be played in reverse order. Looping modes are affected by this function and will start playing / looping the slices in reverse too.

- **Transparent Mode:** When enabled, "Transparent Mode" lets the audio from the inputs to pass through, to the outputs unless a slice is triggered. This function allows you to intervene only when you need to alter normal flaw of the incoming audio.

2.3.1. Roll Buffer Function

You should always consider the Roll function to be a similar to "The Echo" roll function.

Recording some audio, enabling the roll function to keep it, and then saving your song, will *NOT* save the audio data of the buffer, next time you load it!

Always remember that the stored audio, you wish to use when enabling "Roll", has to be previously recorded by BeatChop, at the preceding song playtime. Just like you'd do with a delay!

In a delay unit, setting the feedback to 100% would not save the feedback audio to the song! You'd have to let the delay device to "listen" to the original audio material again in order to repeat the feedback.

2.3.2. Transparent Mode

Transparent Mode function enables you, to intervene to the audio only when necessary, so that you do not have to reconstruct the incoming audio by drawing continuous events in the Key Editor. This is done by fast ducking the incoming audio when a slice is played.

Transparent Mode comes with it's own output level, which you can adjust.

Transparent Mode also is sensitive to the device's ADR envelope. So a slow attack or release will also cause longer fade ins and outs to the incoming audio when ducking from and to "Transparent".

With Transparent Mode set to off, the device remains silent till you hit a MIDI key with an audible slice.

2.3.3 MIDI overriding

All Performance Switches have their dedicated MIDI key overrides. These are a set of four special MIDI keys that override the state of their corresponding

switches (See APPENDIX I). Overriding a switch using its corresponding MIDI key, means that the actual functionality state will be reversed from that of the switch, for as long as the MIDI key is pressed.

So when a function switch is set to on, MIDI override key will set it to off for as long as it is pressed, and vice versa.

Thus the "Roll Buffer" function has a "Roll Buffer" MIDI Key that reverses the state of the "Roll Buffer Switch". "Retrigger Slice" function has a "Retrigger Slice" MIDI Key that reverses the state of the "Retrigger Slice Switch" etc.

This allows the musician to control more of the behavior of BeatChop from his/her MIDI keyboard.

2.3.4 MIDI Slice Divider Keys

The Slice Divider MIDI Keys are a set of four dedicated notes, that divide the length of a slice by 16, 8, 4 or 2 from G#0 to B0 respectively.

The effect takes place for as long as you hold each MIDI key. As soon as you release it, the slice length returns to the actual unaffected value.

3. Front Panel

By now you should have an good idea about the concepts behind BeatChop. So let's take a look at the device itself.

Fig.1.1: The BeatChop Front Panel

BeatChop's Front Panel is separated into 2 separate sections:

- The Main Engine - Upper Unit: BeatChop's Main Engine is a 2U rack mount unit that contains all the parameter controls of the device.

- The DMU-01 Detachable Monitor Unit - Lower Unit: The Monitor is a 1U unit which displays information about the buffer, the output audio level and any MIDI interaction.

BeatChop's Main Engine is divided into 7 different sections:

- **Performance Controls:** (Leftmost) These are your well-known Pitch Bend and Modulation Wheels.

- **Performance Modifiers:** These are situated next to Performance Controls, and include:
    a. Bend Range Dial: To set the size in semitones of the Pitch Bend Wheel
    b. Mod Wheel Targets: 3 knobs that affect the Slice Start Offset, the Slice Length and the Volume respectively.
    c. Velocity Target: 1 knob that affects the Slice Volume.

- **Buffer:** This section handles the buffer offset and size. The first is set using the "Offset" dial, while the second is a product of the "Steps" dial value, multiplied by the step length indicated by the "Resolution" value.

  There is also a LED on top, that warns you when and if the device is downsampling incoming audio to fit the contents to the buffer size.

- **Performance:** This section provides an interface to the Performance Switches discussed in Chapter 2. Each of the four performance

switches comes with a push button and a LED to indicate their state. Under the state LED, lies a second MIDI override LED which informs the user of any overriding done by midi.

What's more "TRANS" (Transparent Mode) switch comes with a knob that can be used to adjust the volume of the "transparent" incoming audio.

- **Slice:** In this section you can define the playback behavior of the slices. The following controls are provided:

    a. **Loop Mode:** Enables or disables the looping for a slice when set to off. A slice can be looped either forward (directional), or forward-backward (bi-directional or alternating).

    b. **Start & Start Quantize Switch:** According to the Quantize switch state, on or off, the Start knob will offset the slice start by a note length divisor, or by percentage respectively.

    c. **Length & Length Quantize Switch:** Likewise to the above, the length knob will either shorten the length of the slice to a note length divisor, or by a percentage.

- **Pitch:** This section controls the pitch of all slices. You have the ability to adjust the pitch in two ways:

    a. Semitones: Use the semitones dial to adjust the pitch in semitones deviation from the original pitch.

    b. Speed: Use the speed knob to adjust the pitch into a linear relative speed from the original.

- **Envelope:** These controls enable you to shape the amplitude envelope for all slices. The envelope generator is a typical ADSR (Attack, Decay, Sustain, Release). The four knobs under the envelope section correspond to each one of these parameters.

- **Output:** The output section offers controls that affect the entire output of the device. These are:

    a. Volume: Controls BeatChop's main volume.

    b. Pan: Controls BeatChop's global pan position.

    c. Mix: Controls the Dry / Wet signal balance for the device.

The DMU-01 Device is divided into three distinct sections

- **MIDI:** The section provides a set of 3 LEDs indicating (from top to bottom) if:

    a. A Slice is triggered
    b. A performance override MIDI key or a MIDI Slice Divider key is pressed
    c. A pitch MIDI key is pressed.

- **Buffer Status:** Provides information about the buffer contents, the recording and the playback activity.

- **Output Meter:** A peak meter that measures the level of the audio output.

3.1. Performance Modifiers

The performance modifiers section is situated next to the Pitch Bend & Modulation Wheel pair.

On top you can find the "Bend Range" dial that adjusts the width of the pitch when using the Pitch Bend Wheel.

The setting can expand from -24 (Inverted Pitch Bending) to +24.

Just Bellow "Bend Range" dial, three knobs are located: "S.Sta" (Slice Start), "S.Len" (Slice Length) and "Vol" (Volume).

These are the Modulation Wheel Targets, with which you can adjust how much will the Modulation Wheel affect each of the three control parameters.

The knobs are bi-directional, meaning that when in the middle position, no modulation will occur. Turning the knobs from the middle to the right will cause the Mod Wheel to *increase* the value of each target parameter by a percentage from its current value. Turning the knobs from the middle to left will *decrease* the target parameter values by a percentage.

Note that summed parameter values (parameter knob plus modulation values) cannot exceed the maximum limits, those of their controls, which means that increasing the slice length f.e. has effect only if the Slice Length knob is lower than 100%.

Next to the three Modulation Wheel Targets is the Velocity to Volume knob that controls the amount of velocity used to affect each slice's volume. At 0% velocity will not affect the volume of a slice. At 100%, velocity will fully affect the volume for each slice. 100% is the compatible setting to previous BeatChop versions.

3.2. The Buffer Section

The buffer section defines the length of the buffer and the options for the slice auto-generation. As we've discussed on Chapter 2 (Concepts), the length of the buffer is determined by the amount of steps multiplied by their resolution.

Lets examine the section starting from the middle this time:

- **Steps:** is where you set the amount of steps that the buffer will record. The parameter can be set from 1 to 64.

- **Resolution:** Resolution is in other words the note length of each step. You can adjust this at any musical time value starting from 1 (whole) and up to 64 (sixty-fourth). There are also triplets and dotted resolutions for each value.

- **Offset:** You can tell the buffer to start recording by an offset number of steps. These steps are taken into account from the start of the song and *not* at the start of the playback. In case "Offset" is greater than the number of "Steps", the offset is trimmed down to the remainder of the division Offset / Steps. For example if Offset = 20 and Steps = 16 then the offset is trimmed down to 20 mod 16 = 4.

3.3. Performance Switches

Performance Switches are a set of functions that can be applied to the processed audio, and they expand BeatChop's performance capabilities.

Each function comes with it's own group of controls. Namely, a button that enables or disables the function, and two LEDs: One for the switch / function state (ON), and one for the override state (OVR).

Overrides occur when you press the assigned MIDI keys for the functions. What a MIDI override key does is that it reverses the state of it's corresponding function, thus if "ROLL" (Roll Buffer) is set to "On", pressing the corresponding MIDI key will disable the function, and vice versa.

The four Performance Switches provided are:

- **ROLL:** Refers to the "Roll Buffer" switch. When set to "On" it stops the buffer from recording, meaning that the old, already written, contents will be retained (kept) and used as slices.

- **RETR:** Refers to the "Retrigger Slice" switch. When set to "On" Slice playback starts at the very first sample (or last if reversed). With "Retrigger Slice" switch set to off, BeatChop will start playing a slice taking into account the delay between the moment the slice was supposed to start if quantized, and the moment it was actually triggered. This is a very powerful feature, especially for gate effects, but also live performances, as it ensures that played slices will be in sync with the song.

- **REVR:** Refers to the "Reverse Slice" switch. When set to "On" all slices will be played in reverse instead of forward. Also if forward looping is used, slices will loop in reverse. Another key point is that "Retrigger Slice" switch state remains in effect. Without retriggering slices will

start at the appropriate offset from the end.

- **TRANS:** Refers to "Transparent Mode" switch. By default this switch is enabled, so BeatChop will output incoming audio except when you trigger slices via MIDI. However there will be times when you want BeatChop to remain silent, and playback only the slices you trigger.

  "Transparent Mode" control group comes with a knob that sets the volume of the passing through audio.

3.4. Slice Section

The "Slice" section provides controls for setting up the playback of the slices. This includes the Start & Length, but also the Looping Mode for all slices.

- **Slice Start:** Changes the start offset of the slices. The value is expressed as a percentage, and it reflects a percentage of the value set by the Buffer Resolution unit.

- **Slice Length:** Changes the length of the slice. The value is also expressed in percentage, but you should bare in mind that the complete length of a slice is the span between "Slice Start" offset and the end of the slice. Therefore a "Slice Length" of 50% applied to a slice with "Slice Start" 50% will play only the 25% of the actual slice.

- **Quantize Switches:** Both "Slice Start" and "Slice Length" can be set to use a set of "quantized" values. Yet, because slices vary according to the "Buffer Resolution" Unit, the quantization steps are expressed as relative fractional values to the "Buffer Resolution". So 1/4 the length of a 1/16 resolution step is 1 / (16 x 4) = 1/64.

  The possible values for quantization are: 1, 1/2, 1/3, 1/4, 1/6, 1/8, 1/16, 1/32, 1/64 and 1/128.

- **Loop:** Sets the loop mode for all slices. Looping occurs -as you'd expect for the span between slice start and up to slice length. There are three available loop modes:

  a. Off: The slice stops when its end is reached and no looping will occur.

  b. Forwd: (Forward Only) The slice will loop continuously in the direction set by the "Reverse Slice" switch (either forward or reverse).

  c. Altern: (Alternating, Forward / Backward) The slice will start looping from the direction set by "Reverse Slice" switch, and

then the other way around.

3.5. The Pitch Section

The pitch section affects the global pitch for all slices. You can alter the pitch in 2 different ways:

- **Semitones:** Use the spin dial to adjust the pitch semitones. You should use this setting to make tonal adjustments to the incoming audio.

  Note that the "Semitones" value does not reflect changes you make in pitch semitones using the dedicated transpose octaves via MIDI.

- **Speed:** Adjust the pitch speed. When you want linear transitions in pitch change (for creating tape stops, scratch-like effects etc). Its better to use the "Speed" knob.

Both of these settings cover 2 octaves up and down each.

3.6. The Envelope Section

A Typical ASDR (Attack, Decay, Sustain, Release) Envelope is provided to shape the amplitude for all slices. Although you can use it in any situation, most of its usability is revealed with when slices are in loop mode.

Sustain at 0% maintains backwards compatibility with the ADR envelope, used in the previous versions of BeatChop.

Note that "Transparent" function "obeys" the ADSR envelope, so a slow attack, decay or release will cause the transparent audio to slowly fade in or out rather than instantly disappearing.

3.7. The Output Section

BeatChop's output section provides three controls:

- **Volume:** Adjusts the global volume of the device.
- **Pan:** Adjusts the global pan positioning of the device.
- **Mix:** Adjusts the weight between the Dry and the Wet signal of the device.

Please note that the pass-through audio from the "Transparent" function is considered as "Wet" signal.

3.8. MIDI Indicators

BeatChop Monitor Unit provides a set of LEDs that serve as MIDI indicators:

- **SLICE:** Lit when a slice is triggered via a MIDI key
- **PMOD:** Lit when a performance switch is overridden via a MIDI key, or when a MIDI Slice Divider key is pressed.
- **PITCH:** Lit when a pitch MIDI key is pressed.

3.9. Buffer Status

The Buffer Status section displays a rough graphical representation of the buffer's contents, along with state information such as the Recording and Downsampling state of BeatChop's Buffer.

The section features 32 "Peak" LEDs, each one representing a step. The LEDs light in variable heights according to the strength of the stored audio. Below them, another set of 32 "Step" LEDs gives information about the state of each Step. In Particular.

- A Blank LED: Indicates a step that is available for recording to the buffer.
- A Yellow Colored LED: Indicates a step that is not available for recording to the buffer.
- A Green Colored LED: Indicates the step that is currently played back by MIDI or CV.
- A Red Colored LED: Indicates the step that is currently written by BeatChop's recording mechanism.

Since the "Step" & "Sample" LED Matrix, can only display 32 steps, and BeatChop maximum step count is 64, only half of the available steps can be simultaneously displayed. On the right side of the Buffer Status section there are two indicators that "tell" you which half is represented by the LED Matrix:

- 1-32: When lit, BeatChop displays steps 1-32 to the Matrix.

- 33-64: When lit, BeatChop displays steps 33-64 to the Matrix.

Note that the set of steps that's displayed is set by BeatChop according to the Record Cursor.

Just above The Matrix, you can see the Buffer State indicators. These indicators are:

- **Recording:** This is lit when BeatChop is actually recording audio. Beyond Buffer Rolling there are some special cases -such as continuous tempo automation changes- where BeatChop stops recording audio. In all such cases this LED will be turned off...

- **Downsampling:** BeatChop's internal buffer memory is always 4 seconds long. This means that it can record 1bar of audio in 60 BPM, 2 bars of audio in 120BPM etc. However there are times when 4 seconds are insufficient to record the requested by the user time span. In such cases BeatChop reduces the writing speed, and therefore the sampling rate of the recorded audio to fit the time span to that 4 seconds. This is called downsampling and the quality of the recorded audio is lower.

  The "Downsampling" LED will lit to warn you about such situation occurs.

3.10. Audio Output Peak Meter

The Audio Output Peak Meter gives you a visual feedback of the audio output of the device.

➘ 4. Back Panel

Figure 4.1: BeatChop Rear Panel

The BeatChop back panel is separated in to two parts:

- **Audio Connections:** These are two stereo outlets for audio input and output.

- **CV Connections:** The device offers a typical Note / CV outlet pair, for sequencer control, and modulation inputs for the following:

    - MW: Controls the Modulation Wheel.
    - Start: Controls the Slice Start Offset.
    - Len: Controls the Slice Length.
    - Speed: Controls the Pitch Speed.
    - Env A: Controls the Envelope Attack
    - Env D: Controls the Envelope Decay
    - Env R: Controls the Envelope Release
    - Mix: Controls the Dry / Wet Signal Balance
    - Pan: Controls the Master Panning position
    - Vol: Controls the Master Volume

Modulation CV Inputs are configured in such way that each polarity can cover the entire controlled parameter.

For example the positive range of the "Pan" CV modifier is sufficient to change pan position from fully left, to fully right, if "Pan" knob is set to fully left. Likewise the same applies for the negative range which is sufficient to change pan position from fully right to fully left, if "Pan" knob is set to fully right.

When the said "Pan" knob is the middle position, it takes only half of the bipolar range (both positive and negative ranges) to fully pan from left to right.

➘ 5. Getting Started Tips & Tricks

5.1. Adding Some Groove To Vocals.

Here is an example of how to add some rhythmic groove to a vocal, ad-lib or even entire acapella. Adding rhythmic grooves can really make a difference, and uplift the vibe of your track:

1. Create an audio track
2. Load a vocal of the same tempo as your song.
3. Synchronize by moving it, and if necessary copy it to create a basic skeleton of where it should sound.
4. Add a BeatChop as insert effect to your Audio Channel.
5. Right click on BeatChop and select "Create Track for BeatChop..."
6. Play one bar of the vocal groove you've just created, and stop the sequencer. Now if you hit any of the slice keys, a short portion of your vocal should be heard.
7. Create a part.
8. In the key editor pick the slices you like the most, and draw a 1 bar pattern of 1/16 notes out of them.

   Remember to prefer lower slices at the start of the bar since these are the ones that will be recorded first.
9. If your vocal is bigger than one bar, repeat steps 6 to 8 for the other bars.
10. Copy if necessary both vocal parts, and BeatChop note parts.
11. Rewind the cursor at the start of your groove and hit play

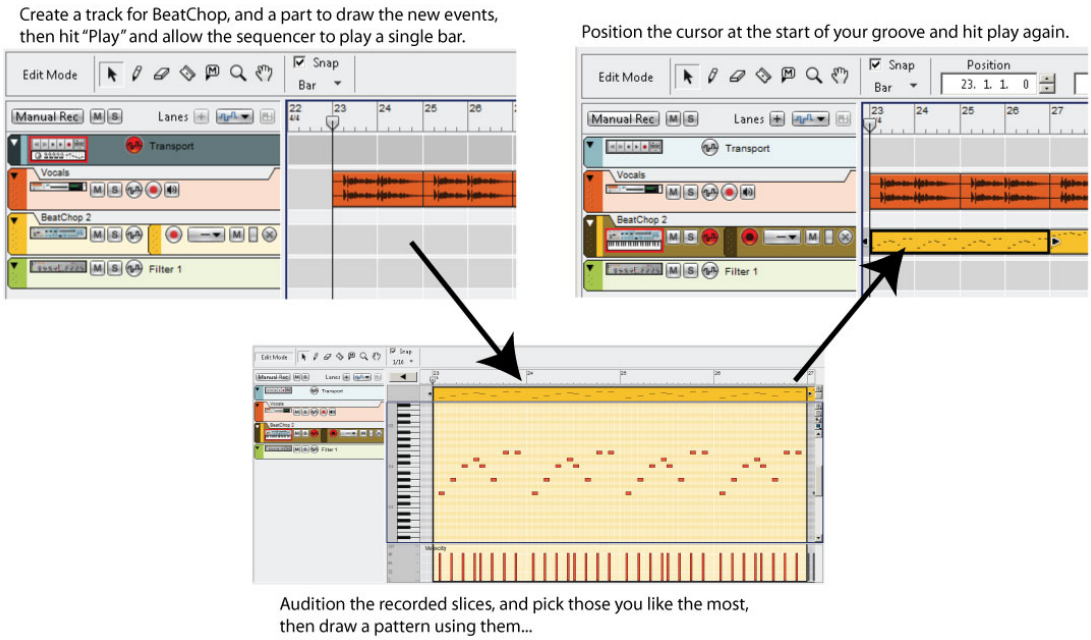


Fig.4.1-1: Adding some rhythmic groove to your vocals.

On-Line Example File:
http://www.quadelectra.com/files/zip/BeatChopExamples/BeatChop_Example_-_Adding_Some_Groove_To_Vocals.zip

5.2. The 'Famous' Guitar Stutter Effect.

It has been one of the hippest and coolest effects for some time now. Even Madonna, couldn't resist and sung over it in "Don't Tell Me". The famous guitar stutter trick, was originally produced by slicing a guitar loop and tampering it, by repeating its parts with slices, and later with dedicated stutter plug-ins.

BeatChop is ideal for this kind of treatment, and can deliver results much more efficiently than sampling / editing and slicing, as well as much more precise handling of the audio material than any other stutter plug-in.

Here's how to do it:

1. Create a loop using either an instrument or recorded audio of a guitar (or any other instrument for that matter). Remember that the key here is to create an arpeggiated chord. Don't just use flat chords! The more rhythmical elements, the better.

2. Perform the steps 6-8 from the previous example, only this time be careful not to overdo it. You can also draw longer slices than 1/16 here.

3. Turn off the "Transparent Mode" switch, and draw some "Transparent MIDI key" events instead to let audio to pass through, from your guitar

riffs, at the most significant parts of your arpeggiation.

4. Enable slice loop mode to forward.

5. Enable Slice Length Quantize, and automation for the Slice Length (Quantized) parameter, and shorten its value, right under some of the slice notes you played.

6. Now play the phrase. You may need to repeat these steps, until you are satisfied.

On-Line Example File:
http://www.quadelectra.com/files/zip/BeatChopExamples/BeatChop_Example_-_The_Famous_Guitar_Stuttering.zip

5.3. Tape Stop & Scratching.

Tape Stop (or Record stop) is a well established effect from the old days of multichannel tape recorders and mastering tapes. Sound engineers would stop a track from playing on a tape recorder, record the actual pitch / speed drop and use it as a fill-in, as part of the track it self.

With BeatChop you can do this effect in real-time, without recording / sampling anything:

1. Add a BeatChop at the end of your master buss. You want this effect to process the your mix after your master effects, so route it after any master insert effects.

2. Change the Buffer settings to 4 steps, of resolution 4 (one quarter). Alternatively if you want longer tape stops, you can set the buffer to 2 steps of resolution 2 (half), or even 1 step of resolution 1 (one bar).

   In this example we want a fast tape stop so the first setting of 4 quarters will do just fine.

3. Add a note lane (Track) to BeatChop if its not added already.

4. At the end of your phrase, create a part at the size of a bar for BeatChop note lane, and draw an event of one quarter at the last beat of your phrase, at note D#2.

   Thats the fourth slice, that corresponds to the last beat. Actually any slice MIDI note will do, assuming there was at least one preceding bar of audio, but D#2 is the slice at which your underlying audio will be recorded at that time.

5. Enable the automation for the "Speed" knob under the "Pitch" section.

6. On the opened "Speed" automation lane, draw a point at the start of the MIDI quarter key event you created, at the last beat on bullet #4.

   Make it so, so that the automation point is above the middle value (no pitch). That is, set the speed a bit higher at start.

7. Then at end of the part and the phrase, draw another point, this time at the bottom of the automation lane. To make the drop, a tad more punchy, at the second 1/16th of the drop start add another point and drag it down a bit to "skew" the linear fall ramp of the pitch.

8. Press play about a bar before the end of your phrase to allow BeatChop to record some audio, and enjoy!

   Bare in mind that you can even make tape rises (increasing the speed), and even scratch effects if you use shorter (about 1/16 resolution) and reverse mode, along with pitching up and down your events. The example file below shows some of these techniques.

Fig.4.3-1: That should do it! Notice that the Speed automation ramp is slightly skewed down at the second step

On-Line Example File:
http://www.quadelectra.com/files/zip/BeatChopExamples/BeatChop_Example_-_Tape_Stops_Rewinds_Scratches_Plus.zip

5.4. More Advanced Tricks for Playful Vocals.

Sometimes you want to add some "playful" elements to the vocals of your track, such as stuttering, or glitching. By combining the tips from previous examples you can achieve such results on vocals. But what if you want a glitch to proceed the actual vocal from which it originated?

Here's how to do it:

1. Place your acapella or stem to the sequencer. If it begins at first bar move your song to bar #2.

2. Copy the audio track and move the part (or parts of the new copy) one bar earlier.

3. Create a BeatChop at the Audio Track Copy.

4. Turn off "Transparent Mode Switch". Now BeatChop will output any audio when you hit a slice.

5. Start working like in Example 4.1. and create some events, the strength of which will escalate at the end of the bar.

   For example draw 1/16 notes of the first or second slice on the last 2 beats of the bar, and use the line tool to increase velocity.

   Alternatively, draw one single note at the size of 1/2 for the last two beats, shorten the Slice length about 20% to 25% and increase the Env Amount (better yet: Enable automation and increase the Env Amount for that bar only by drawing a point).

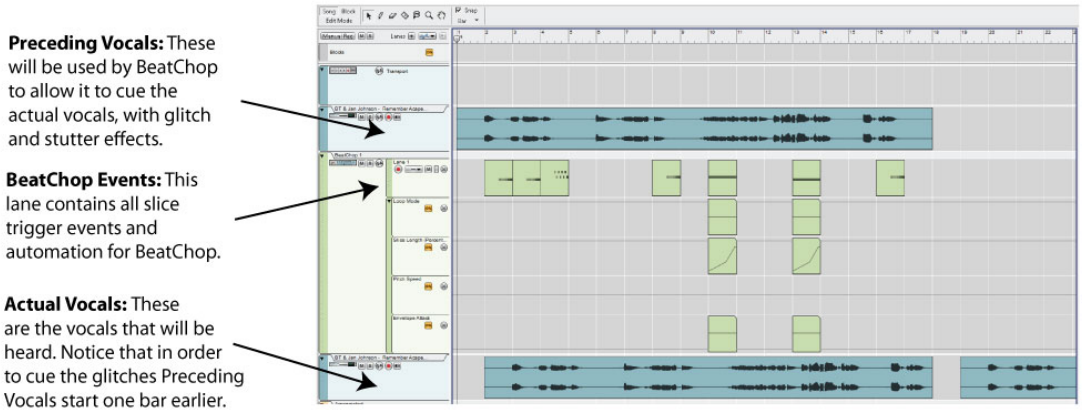


Fig.4.4-1: The arrangement with the actual and preceding vocals on which BeatChop will operate.

6. Listen to the result, and repeat the process as needed.

Here's a way to create a nice gate effect, that uses random slices from your audio material:

1. Add a BeatChop to the target channel.

2. Select a buffer resolution and step count. Don't be frugal! 32 steps of 1/32ths or even 64 steps of 1/32ths f.e. is a good selection.

3. Turn off transparent mode.

4. With the BeatChop selected (highlighted), click Right mouse key (Command +

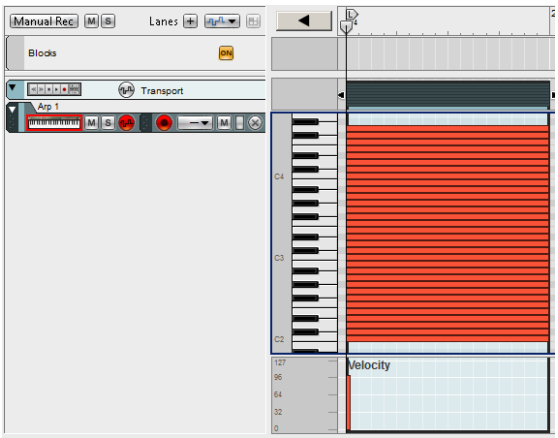


Fig.4.4-2: A Stack Of 32 MIDI events of one whole each. These will be [illegible] enough to trigger [illegible]

Mouse Key on Mac) and from the context menu select Utilities -> RPG-8 Monophonic Arpeggiator. Verify that BeatChop and the RPG-8 are connected via Sequencer CV note / gate.

5. At the RPG-8's note track, create a part one bar long, and draw a series of stacked successive midi events, one per semitone, starting from C2 (first BeatChop Slice), for as many steps as you've set your buffer to. If f.e. you have set the buffer to 32 steps draw a stack of 32 whole notes, one per semitone, in your bar.

6. At the RPG-8 choose "Random" as mode.

7. Copy the bar with the stacked notes you've created, for as long as your audio track plays.

8. Hit Play and enjoy. Moreover you can shorten the Gate Length, to achieve staccato sounding results.

On-Line Example File:
http://soundcloud.com/quadelectra/beatchop-demo-vocals-glitching (Audio Only)

I'm not hearing any audio.	o Check your connections. o Have you allowed BeatChop to actualy "hear" and record some audio first? o Have you drawn some MIDI events on BeatChop's note track? o If you are using a keyboard, make sure you're playing some keys at the right range (C2 - D#7)
I don't see a Note Track for BeatChop.	Sometimes Reason creates BeatChop instances in the sense of an effect. You'll need to create a Note Track your self by Right Clicking (Command + Click on Mac) and selecting "Create A Track For BeatChop" from the context menu.
I hear clicks in the audio.	o BeatChop Attack, Decay & Release slopes can get very steep in order to be able to work with sharp, punchy sounds. However in long flat sounds like drones and pads a steep attack or release may produce audible clicks. Try loosen up your Attack and Release settings. o Buffer Rolling might also cause clicky artifacts, since sometimes it may be disabled or enabled a tad earlier or later than the actual slice start, leaving a short trail of the old or new material. Enter edit mode, disable quantize, and try moving the automation point a bit to the left or right depending on the situation. Note that this solution is recommended for audible clicks in loops.
I change the pitch, but it doesn't work!	In order to avoid buffer overlapping (meaning playing back data where the buffer is recording), BeatChop has a safety switch that won't allow pitch changes higher than normal in the slice currently recorded. There is a list of rules when this applies. *See Section: 2.2.4 "Safe" Mode*

My issue or problem is not listed here. What should I do?	Feel free to contact us about ANY problem you may have at: audioworx-support@quadelectra.com

APPENDIX I: MIDI Key Assignments

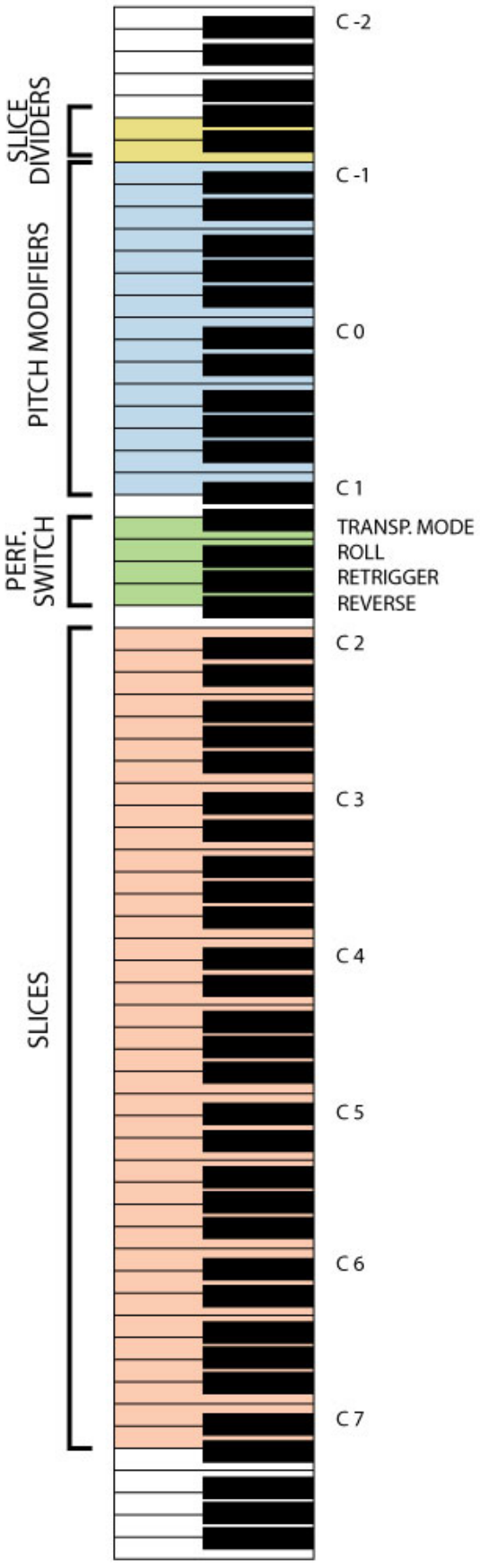
The chart at the right shows BeatChop's MIDI Key Assignments for all available functionality.

In particular:

- G#0 - B0: Slice Dividers. B0 = 1/2 of Slice Length, A#0 = 1/4 of Slice Length, A0 = 1/8 of Slice Length, G#0 = 1/16 of Slice Length
- C-1 - C1: Pitch Semitones (C0 = unmodified pitch)
- E1-A1: Performance Switches, E1= Transparent Mode, F1 = Roll Buffer, G1 = Retrigger, A1 = Reverse.

  Note: The Performance Switch MIDI keys act as overrides to their corresponding panel buttons. Meaning that they will reverse their state.
- C2-D#7: Slices. These are the 64 actual MIDI keys for the maximum of the available slices.

  Note: The 64 keys will always trigger a slice. (See: 2.2.6 Auto-mapping)



➘ APPENDIX II: MIDI CC Map

The following is a MIDI Control Number Chart for all available BeatChop 1.0 functionality

CC #	Description
7	Main Volume
8	Dry / Wet Mix
10	Pan
12	Slice Start (Percentage)
13	Slice Length (Percentage)
16	Slice Start (Quantized)
17	Slice Length (Quantized)
18	Slice Start Quantize Switch
19	Slice Length Quantize Switch
20	Slice Loop Mode
52	Buffer Offset
53	Buffer Steps
54	Buffer Resolution
62	Pitch Semitones
63	Pitch Speed
65	Roll Switch
67	Retrigger Switch
68	Reverse Switch
69	Transparent Switch
70	Transparent Volume
73	Envelope Release
72	Envelope Attack
75	Envelope Decay
79	Envelope Sustain
85	Pitch Bend Range
86	Mod Wheel to Slice Start
87	Mod Wheel to Slice Length
88	Mod Wheel to Volume
89	Velocity to Volume

↘ TABLE OF CONTENTS