A Complete Guide to

Manual v2.0
Welcome to Plogue chipcrusher

Plogue chipcrusher has been always been an odball in our 'chip' line - which mainly consists of synthesizers. Each time we found something cool in our chip research that would only make sense in the context of an FX, we just basically just threw it in there. So in this context, you need to forget its name, its now much MUCH more than a 'bitcrusher'

This new updated version contains bits of research from chipspeech (MOZER/SP0256 encoding), a bit accurate 32kHz SPC Delay effect from our forthcoming “chipsynth SPC” synthesizer, a VOCODER(!!??) and much more...

I don't think there is anything like this quite honestly. Sure there are other 'Bit Crushers' out there, but nearly all of these only emulate the staight PCM sound encoding (we offer 16 distinct ones). There are other Speaker IR libraries, but nothing that covers classic gaming hardware, monitors, and vintage synthesizer speakers.

Please enjoy, and thanks for supporting our continuing crazy research,

David Viens
November 2018
What is chipcrusher?

Plogue chipcrusher is an audio effect which plays back input audio through models of 16 different lofi digital audio encodings (DAC Encoding), adds grit (Background Noise), applies an early lofi delay effect (SPC Delay) and outputs through a selection of speaker and filter impulse responses (Cabinet). Since there's a wide spectrum of results achievable with chipcrusher, we've targeted a few uses:

1) You just want to uniquely destroy/mangle a beat, a guitar or any other audio track.
2) You want to play single hits and emulate the sound of old samplers.
3) You want to add 'accurate dirt' to other plugins' output. Quite a few users have told us that the outputs of our other 'chip' products were too 'clean' and would have liked some of the characteristic dirt present in recordings of original consoles. For this particular case, make sure that the dry audio output is sent to chipcrusher, and turn DAC Encoding off.

Plogue chipcrusher can work as a plug-in for most major sequencing audio programs.

There is a misconception that if you send any audio track into a bit crusher that it magically makes it '8bit'. This is neither true, nor the purpose of this plugin. The R&D behind this product was quite significant. We had to modify each and every device included one way or another to get the required impulse responses, noise, and test recordings.

What’s Included

This version of chipcrusher includes the following:

- Download file containing the complete chipcrusher software.
- The unique graphical license key for the product.

Before you begin installation, make sure you've read the End User Licensing Agreement in the following pages. By installing the software, you indicate you agree to the terms of the license.

Further Documentation and Resources

For the latest information including updated documentation, visit our support pages at: www.plogue.com. There you can find updated information provided after the manual was written, corrections or additions to this manual FAQ pages answering common questions suggestions from the users of Plogue software and news about upcoming Plogue releases. You can also visit the Plogue Forums for up-to-date information.

The address is: www.plogue.com/plgfrms.

Please send any reports of errors in this manual or suggestions for improvement to: chipcrusher.support@plogue.com
End User License Agreement

Please read the terms of the following software licensing agreement before using this software. By installing and loading this product on your computer you acknowledge that you have read this license agreement, understand the agreement, and agree to its terms and conditions. If you do not agree to these terms and conditions, do not install or use the sounds contained herein. This is the complete agreement between you and Plogue Art et Technologie, Inc that supersedes any other representations or prior agreements, whether oral or in writing.

An important thing to understand is that YOU ARE OBTAINING A LICENSE FOR YOUR USE ONLY—THE NOISE RECORDINGS and IMPULSE RESPONSES do NOT belong to you. The implications are described below. The sounds, samples and programming in the Plogue chipcrusher program remain the sole property of Plogue Art et Technologie, Inc and are licensed (not sold) to you.

What You Can Do: You can use these effects in music productions, public performances, and other reasonable musical purposes within musical compositions. You can use these effects in your own musical compositions as much as you like without any need to pay Plogue Art et Technologie Inc or obtain further permission. If you do use these effects, we kindly ask that in any written materials or credits accompanying your music that utilizes material from Plogue chipcrusher (CD booklet, film credits, etc), that you include the following courtesy credits: “Some sounds were processed through Plogue chipcrusher”, or a similar credit where practicable. If you can’t disclose your use of chipcrusher for any reason to the general public, we would at least love to be aware of the usage of chipcrusher in a commercial recording for our own PRIVATE records.

What You Cannot Do: The enclosed noise recordings and impulse responses cannot be re-used in any other commercial sample library or any competitive product. You are absolutely forbidden to duplicate, copy, distribute, transfer, upload or download, trade, loan, reissue or resell this product or any of the contents in any way to anyone. You cannot redistribute them through an archive, nor a collection, nor through the Internet, nor binaries group, nor newsgroup, nor any type of removable media, nor through a network. The sounds and samples contained herein cannot be edited, modified, digitally altered, re-synthesized, or manipulated without direct written consent of Plogue Art et Technologie Inc.

Disclaimers and Conditions: A right to use Plogue chipcrusher is granted to the original end-user only, and this license is not transferable unless there is written consent from Plogue Art et Technologie, Inc and payment of an additional fee. The sounds of Plogue chipcrusher will only work with the bundled Plogue Fermata Engine and will not work with any other engine. Licensor will not be responsible if the content does not fit the particular purpose of the Licensee. Please make sure before installing this item that it meets your needs. Information contained herein is subject to change without notice and does not represent a commitment on the part of Plogue Art et Technologie, Inc. The sounds are licensed “as is” without warranties of any kind. Neither Plogue Art et Technologie, Inc, nor any agent or distributor can be held responsible for any direct or indirect or consequential loss arising from the use of this product in whatever form. The Fermata Engine is covered by the installer’s End User License Agreement and is incorporated by reference. Plogue chipcrusher may not be returned for any reason. The terms of this license shall be construed in accordance with the substantive laws of Canada and the province of Quebec. The user agrees to read the manual before seeking technical support and to make sure his or her system meets the recommended requirements.
Specifications & Computer System Requirements

The following table lists the computer and hardware requirements for using chipcrusher:

- Plogue chipcrusher v2 is **64-bit only**.
- You can use chipcrusher on almost any modern personal computer that meets the specifications listed below.
- The specifications provide the minimum standards.
- For optimal functioning, it is recommended that you have a powerful enough computer with a fast CPU processor.
- The stated requirements are an average. Certain music program may have additional resource requirements.
- Please also observe the systems requirements of your host application, tracker program and/or sequencing program if applicable.
- The demands of various other processing software (including the sequencer, audio and effects processors, other plug-ins, and so on) can affect functionality.
- Please see the Plogue forum or website if you are looking for recommendations or for more information.

<table>
<thead>
<tr>
<th>Computer</th>
<th>Operating System</th>
<th>Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Windows PC (64 bits)</td>
<td>Microsoft Windows 7 (64 bits)</td>
<td>✦ 64-bit CPU, Dual Core or better</td>
</tr>
<tr>
<td></td>
<td>Microsoft Windows 8 (64 bits)</td>
<td>✦ 64 MB of free hard drive space</td>
</tr>
<tr>
<td></td>
<td>Microsoft Windows 10 (64 bits)</td>
<td>✦ Internet connection (for downloads and updates)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ Monitor with 1280 x 1024 resolution or better</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ A suitable VST/ host, or Pro Tools 12 or later</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ Keyboard: A MIDI interface may be required if you are using a MIDI keyboard. Some keyboards use USB.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ High quality speakers and amplifier, or high quality headphones</td>
</tr>
<tr>
<td>Mac (64 bits)</td>
<td>Mac OS X 10.9 minimum (64 bits)</td>
<td>✦ Mac Intel 64-bits; Dual Core or better recommended</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ 64 MB of free hard drive space</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ Internet connection (for downloads and updates)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ Monitor with 1280 x 1024 resolution or better</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ A suitable AU or VST host, or Pro Tools 12 or later</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✦ High quality speakers and amplifier, or high quality headphones</td>
</tr>
</tbody>
</table>
INSTALLATION

**Installing Plogue chipcrusher**

Installing Plogue chipcrusher is easy. Before you begin, make sure you meet the system requirements. The full installation requires approximately 64MB of free system hard disk space. Installation involves installing the chipcrusher plugin and also the FERMATA engine. A setup program will guide you through the process step-by-step.

**Mac Setup**

- First, make sure your audio and MIDI hardware is set up and working with your computer.
- Close any programs you are running.
- Double click on MAC_chipcrusher_v2.xxx.pkg (Accept the defaults on the installer. You should see a Welcome Screen.)
- You’ll be asked to accept the End User License Agreement to proceed with installation.
- The setup lets you select which plug-in formats you need, so that chipcrusher can run as a virtual effect, seamlessly integrating into your favorite plug-in supporting music software; the choices are Standalone, Audio Units (AU), AAX and VST (must select at least 1).
Setup is now ready to begin installing Plogue chipcrusher. Click on “Install” to begin installation and the program will begin installing. This process should not take more than a minute.

After a few moments a dialog box will notify you that the setup wizard has finished installing Plogue chipcrusher on your computer.

Important:
Please do not cancel setup after installation begins, otherwise a partial, broken installation may result.
Windows Setup

♦ Close any programs you are running.
♦ Double click on WIN_chipcrusher_v2.xxx.exe (Accept the defaults on the installer. You should see a Welcome Screen.

♦ You will then be asked to accept the End User License Agreement before proceeding with installation.
♦ And choose which location to install it on:
- The setup lets you select which plug-in formats you require. These plug-in formats allow Plogue chipcrusher to run as a virtual effect unit that seamlessly integrates into your favorite music software program or sequencer (assuming that it accepts such plug-ins).

For windows, the choices are AAX and VST 64 bit). At least ONE needs to be chosen. You'll be asked to select your 64bit VST plugin folder.

- For the VST plugin install, chipcrusher will ask you what folder to install the plugin in. This must be the folder that your host scans to find VST plugins.

- Please note: chipcrusher v2 is 64bits only
Setup is now ready to begin installing Plogue chipcrusher. Click on “Install” to begin installation and the program will begin installing. This process should not take more than a minute.

After a few moments a dialog box will notify you that the setup wizard has finished installing Plogue chipcrusher on your computer.

Important:
Please do not cancel setup after installation begins, otherwise a partial, broken installation may result.
**Authorizing chipcrusher**

Plogue chipcrusher needs to be authorized in order to be fully functional, otherwise it will run in **DEMO** mode.

When you order a license from Share*it, you receive a personal Activation Key card named *xxxx_key_1015.png* as an email.

The Activation Key card is an image resembling a typical credit card. This image contains your registration and details encoded within the Key card image. It will look like this:

You should save the *xxxx_key_1015.png* image file to your hard drive (keep it in a safe place). For convenience, we recommend that you initially save the .png file to your desktop. You will also receive a copy of the license card in your email inbox.

1) Locate the “license card” image where you saved it on your hard drive. (*xxxx_key_1015.png*)

2) Launch your favorite host and make sure you see chipcrusher interface.

3) Simply click and hold on the file, drag the “license image” or file icon directly onto the application’s UI itself, and release it.

If you don't get any message (or are not able to drop the key in that host), try to *import* the png file from the snapshot load menu at the top right of the interface (use *.* as file filter)
In the rare case when all this isn't working, use the **Import License** button located on the **Settings** page. See the settings page section of this manual.

If none of those methods work, please contact [chipcrusher.support@plogue.com](mailto:chipcrusher.support@plogue.com) and attach your key.

**Extremely Important!!**

The `xxxx_key_1015.png` file contains your **sensitive personal information**, encrypted inside, including your **full name and address** taken from the online shop. Carefully protect this file. **DO NOT GIVE THIS FILE TO ANYONE OR DISTRIBUTE IT IN ANY WAY OR YOUR PERSONAL INFORMATION WILL BE COMPROMISED.** IF THE FILE BECOMES PUBLIC THE CARD NUMBER WILL BE BLACKLISTED AND THE CARD REVOKED. **WE ARE NOT RESPONSIBLE IF YOU GIVE YOUR PERSONAL DETAILS TO A THIRD PARTY. IF THE CARD IS STOLEN, CONTACT US IMMEDIATELY.**

Without a valid card you will also not be able to obtain critical updates to the program.

**Important Note:**

If you have special circumstances or require site licensing, please contact us.

**Updating to the Latest Version**

Be sure to check the Plogue Web sites for any possible updates that have occurred since the time your version of the software was released. Software is frequently updated and a more recent version may be available.
How to Use chipcrusher

Once installed and authorized, it’s time to get started with Plogue chipcrusher. Just choose the chipcrusher plug-in in your favorite host, typically on a stereo track as an insert:

There are two components to pre-condition the sound (Gate in/out, Compressor) followed by the four main components (DAC Encoding, Background Noise, SPC Delay and Cabinet).

The next section will explain those components in detail.
**Gate in & Gate out**

This allows for cutting out the distorted fading tails and cutting out the background noise.

In addition to the on/off switch, both gates have the following settings:
- **Threshold**: Level under which the signal is cut out.
- **Ratio**: Ratio between input level and volume drop.

**Compressor**

Using the compressor, you can equalize the input level of the sound going into the DAC, allowing for a more even effect. It can also be used for special effects. The compressor includes a make-up gain (increases when threshold is lowered or ratio is increased).

- **Threshold**: Level over which compression starts to be applied.
- **Ratio**: How much compression is applied when the compressor is active.
- **Attack**: The time it takes for the compressor to reduce the volume when the input gets louder.
- **Decay**: The time it takes for the compressor to increase the volume when the input gets softer.
- **Rms Time**: The overall sloppiness in the way that the compressor tracks the volume.
- **Parallel**: Balance between the compressed sound and the unprocessed sound.
- **Volume**: Gain applied before sending out to the rest of chipcrusher.
- **Multi-Band**: Balance between compression of the whole signal, and compression in 4 bands.
- **Band balance**: Balance between low and high bands when using multi-band compression.
Plogue chipcrusher’s **DAC Encoding** lets you do the following passes on the input sound:

1) Resample the input audio input to a reduced sample rate.
   - **Pre Gain**: Gain applied before sending the input into the DAC Encoding. This lets you adapt the level to match with what the encoding works best with – or clip the signal on purpose!
   - **Resample**: Reduces the sample rate to this frequency.

2) Encode the data using a vintage data compression algorithm (See **ENCODINGS explained**).
   - **Encoding**: Vintage data compression type and DAC type used.
   - **Bits**: Bit depth used by encoding. The results are highly variable depending on the type. Some encodings do not use this setting.
   - **Extra parameter**: Some encodings let you tweak extra aspects, such as having multiple specialized sub-modes, or having multiple bit depths. Not all encodings have this.
   - **MIDI in**: (Vocal encoders only) Enables the various pitched components of the vocal encoders to be controlled using live MIDI input instead of a fixed frequency only. You must be playing notes for sound to come out!
3) Change the weighting of bits in the DAC to emulate non-monotonic behavior.

**Monotonicity:** When monotonicity=100%, all the bits of the DAC influence the output as perfectly as you'd expect. When you drift from that, you get weird waveshaping artifacts due to bits slightly changing 'weight'; electronically speaking, if you think of a 'Resistor ladder' type DAC, that the resistances are progressively going out of tolerance. Values of 95-100% should be used to stay in the 'realistic range', but unrealistic values are very interesting!

4) Return to the host sample rate using vintage DAC behavior. (Zero-Order Hold, PWM, etc)

5) Apply basic filtering: one pole low pass (RC Filter) and high pass (AC Coupling). These basic filters are extremely common in the output sections of all kinds of electronic devices.

RC-Filter: Cutoff of the low pass filter.
RC-Filter Bypass: Disables the low pass filter.
AC Coupling: Cutoff of the high pass filter.
Post Gain: Volume after the DAC section. Still applied when the DAC Encoding section is off.

**Simple Encodings**

![PCM](image)

The simplest of all encodings. The sample is simply reduced down to the desired number of bits.

**Bits:** How many bits of the sample are kept.
**Delta Bits:** How much the steps are offset from 0. This acts as a sort of gate at low bit depths.

![DPCM](image)

This encoding uses the difference between samples instead of the samples themselves, which is a good representation for natural sounds. The constant change in value will add a carrier frequency. Set bits/d.bits to 6 and 1 to simulate a well known classic 8 bit system's DMC!

**Bits:** Bits used for the total value range. Increasing this reduces step size.
**Delta Bits:** Bits used for the step size. This increases the maximum step size.

![FPCM](image)

This adds variable volume scaling to PCM, using finer steps on softer signals. Classic FM synth DACs can be done by setting bits and exponent to 10 and 3.

**Bits:** How many bits are kept at maximum volume. (2-15)
**Exponent:** Volume scaling bits (1-4). This allows the precision of the maximum volume to be kept down to -12dB, -24dB, -48dB and -96dB respectively.
Adaptative Encodings

**BRR (SPC ADPCM)**

This encoding groups samples in blocks of 8, and adds a volume scaling factor for the block. This is very similar to FPCM with an exponent of 4. In chipcrusher, the 'no IIR filter' block type is always used (similar to Konami games).

**Bits:** Number of bits kept at maximum volume. Set to 4 for the classic sound.

**A-Law**  
**μ-Law**

These are 8bit telephonic encodings, where the value table has more entries closer to 0, to give better sound at lower volumes, equivalent to FPCM with bits/exponent set to $6 : 3$. DACs using μ-Law were also used on the classic LINN Drum machines, and the E-MU Emulator I/II/Drumulator.

**DIALOGIC (OKI ADPCM)**

This vintage encoding combines 4bit DPCM and FPCM by saving the step from sample to sample but also dynamically varying the step size range, resulting in very good quality! It is fixed, and very closely related to IMA ADPCM. It was used in a LOT of late 80's and early 90's arcade games in the form of the OKI 6295 chip - in particular very famous CPS1 games (before Q-sound).

**YADPCM (Yamaha ADPCM)**

A nice-sounding ADPCM encoding. Similar to Dialogic, this fixed 4bit encoding uses variable step sizes too. It shows up in the N's w'mote and various Yamaha FM chips (for drum samples).

**HAAR Transform**

The Haar transform transforms the sound into overlapping triangle waves. The result is similar to a bitcrusher, but it applies more bit crushing to high frequencies.

**Bits:** Overall bit depth used for the compression.

**CVSD**

CVSD (continuously variable slope delta modulation) is an early 1bit version of ADPCM. This encoding can even be decoded analogically. It was used in encrypted military radios and also famously used to encode Suzanne Ciani's voice in the Pinball classic 'Xenon'.

**Bits:** Tweaks the algo from crunchy (1bit) to smooth (12+bit). 13-16bits enables 2x/4x bitrate.
1bit Density Encodings

**Loud and Piercing 11kHz Carrier Tone Warning!**

These encodings are based on modulating an extremely high carrier tone, which is a violent whine when its frequency falls into the audible range.

Plogue chipcrusher, from version 2 onwards, now has a notch filter designed to remove this frequency and prevent exposure when switching encodings, which defaults to ON. This can be turned off by activating the **PWM/PDM Carrier** option in the settings menu, but this is not for the faint of heart!

---

**PDM**

Pulse-density modulation is a 1bit encoding that continuously varies the threshold between the high and low level, so that the average level of the output matches the input, so that after heavy filtering, the output roughly matches the input. Advanced versions of this using very high sample rates are used in high accuracy DACs (since 1bit DACs cannot have distortion). This is a fixed encoding.

The carrier tone frequency is at $\frac{1}{2}$ the sample rate.

**PWM (Pulse-width modulation)**

This encoding emits a high rate pulse wave, and varies the width of the pulse to recreate the original waveform once filtered. It is commonly used in hardware like microcontrollers, since it requires no active analog components (which are expensive) and no special digital logic.

**Bits:** The bit depth of pulse width. This selects the available number of widths. (1-6bits only)

**Copy:** This multiplies the carrier pulse wave frequency, which makes it easier to filter and helps taking it out of the audible range. (1-6, high values reduce the number of bits)
Vocal Encodings

MIDI Pitch Control

Vocal encodings in chipcrusher support the use of MIDI input to override the pitch detected by the algorithm, allowing for easy robotic musical effects. Whenever a note is played, it overrides the pitch. Turn on MIDI in and try holding a low key for a great menacing robot effect! Pitch bend and mod wheel are supported.

LPC-10 (Texas Instrument LPC)

This is the sound of classic Texas Instruments speech chips based on LPC (Linear Predictive Coding). This uses a complex math algorithm to automatically compute the best matching resonant filter bank. It also uses a pitch detector to generate the appropriate voiced tone and to switch to noise in voiceless segments. This dynamic combination of buzzy tone, noise and filter bank recreates voice-like input signals fairly accurately. This is an extremely compact encoding, since no actual signal is transmitted. It has a very characteristic tone. It was used in the famous Speak & Spell, in addition to the TI 99/4A speech module, and many other toys and speech add-ons.

MIDI in: Lets MIDI notes played override the pitch created. Note that the pitch will be highly approximate due to the limitations of this old hardware!

Mode:
1) Normal (default)
2) Pitched only
3) Noise only

SP0256 (GI LPC)

A buzzy voice compression algo. This encoding is very similar to LPC-10, with even crazier maths going from autocorrelation to quantized and compressed synthesis settings. The tone generation does a full filter reset on each period, giving it a more buzzy tone. It was used in the Intellivoice (Intellivision speech module), the Magnavox Odyssey²'s The Voice module and other similar consumer products based on the SP0256-AL2 chip.

MIDI in: Lets the pitch be overridden by the currently played note.
MOZER

This recreates the TSI and Digitalker line of speech synthesis chips, based on the pioneering work of Dr. Forrest Mozer using heavily reduced waveforms. The frequency can be variable, or fixed for a robotic effect!

The bit depth is dropped to 2 bits (using a DPCM variant), wave periods are repeated, wave length is halved by removing phase information and mirroring the wave, and less active half of the wave is clipped, reducing 128~384 samples down to 32.

This was used in games such as Bezerk, Impossible Mission and Mito Koumon, and in talking calculators and Chess. The encoding process was lengthy (including much manual labor!), but the streamlined decoding meant that it could even be decoded in software on 8 bit processors.

**MIDI in:** Allows the notes to override the current note output.

**Mode:** Selects which data reduction tricks are used:

<table>
<thead>
<tr>
<th>Variable pitch mode</th>
<th>Fixed pitch mode</th>
<th>Multiple wave repeats</th>
<th>Preserved wave length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9</td>
<td>No</td>
<td>Full</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>No</td>
<td>128</td>
</tr>
<tr>
<td>3</td>
<td>11</td>
<td>No</td>
<td>Half</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>No</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>13</td>
<td>Yes</td>
<td>Full</td>
</tr>
<tr>
<td>6</td>
<td>14</td>
<td>Yes</td>
<td>128</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>Yes</td>
<td>Half</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
<td>Yes</td>
<td>64</td>
</tr>
</tbody>
</table>

**Vocoder**

The venerable vocoder has the distinction of being the first voice compression system, used in the SIGSALY voice encryption system in the last stretch of WW2. The original terminal weighted over 50 tons and used 30kW of power. After the war, much miniaturized versions were soon produced, especially thanks to the introduction of transistors, leading to being used in music thanks to pioneering musicians such as Bruce Haack, Wendy Carlos and of course Kraftwerk.

The chipcrusher vocoder has 20 bands, with a built-in carrier synth and a high frequency bypass.

**MIDI in:** Switches off the constant carrier wave and switches on the full midi-controlled synth. This mode is polyphonic and supports MIDI pitch-bend and Mod wheel.

**Mode:** Selects which waveform combination is used on the carrier synth. Various combinations of saw wave are available, in addition to a few pulse and noise-based options.
This recreates the sound of the SPC’s delay effect, which is responsible for the wash-like quality that the SPC gives to orchestral soundtracks such as in Final Fantasy 3. Another well known use is the cavern effect in Super Mario World and Donkey Kong Country. It is also sometimes misused – particularly in fighting games that used a very short length and high feedback to try to make a reverb effect (in vain).

What gives this effect its particular sound – its digital twang - is that the delay time is very constrained (since there is only 64kb of sound ram – shared with all sfx, music, and sound code). It also lacks diffusion (present on later units that could do reverb such as on the PSX or the AWE64). It works very well with instruments with a bit of vibrato (oboes, violins, etc.).

This delay unit has a simple filter (the so-called “8-tap FIR”), that allows for some shaping the frequencies of the reverberation. chipcrusher will automatically calculate the 8-tap filter from the gain at 0kHz, 4kHz, 8kHz, 12kHz and 16kHz, making sure no frequency is amplified.
For people who really want to explore the full range of stable 8-tap filters, the phase shift of each frequency band can also be edited (the shift at 0kHz and 16kHz can only be inverted, due to mathematical constraints). The unstable range of filters is also available by activating the “explode” switch. **Warning: this is very, very loud!**

The Dry, Wet, Feedback and delay Time parameters are calculated as follows for each channel:

<table>
<thead>
<tr>
<th></th>
<th>Left Channel</th>
<th>Right Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dry</td>
<td>$\text{Dry} \times \text{DryInvert} \times \text{DryPan.L} \times \text{DryInvLeft}$</td>
<td>$\text{Dry} \times \text{DryInvert} \times \text{DryPan.R}$</td>
</tr>
<tr>
<td>Wet</td>
<td>$\text{Wet} \times \text{WetInvert} \times \text{WetPan.L} \times \text{WetInvLeft}$</td>
<td>$\text{Wet} \times \text{WetInvert} \times \text{WetPan.R}$</td>
</tr>
<tr>
<td>Feedback</td>
<td>$\text{Feedback} \times \text{FeedbackInvert}$</td>
<td>$\text{Feedback} \times \text{FeedbackInvert}$</td>
</tr>
<tr>
<td>Time</td>
<td>$T \times 512 \text{ samples (16ms)}$</td>
<td>$T \times 512 \text{ samples (16ms)}$</td>
</tr>
</tbody>
</table>

(Pan law = 0dB)

By using the left channel inversion switches, a crude version of stereo widening can be done, but this is generally not a good idea, as the sound will cancel out when downmixing to mono, and the result is often very hard on the ears.

This effect operates at 32kHz – including the dry signal if turned on!
Noises were carefully recorded and looped from our collection of gear (consoles and arcade boards) which were also used for the research behind chipsounds. Just select the preset and adjust volume to taste. Note: default volumes should be used if you want to keep it subtle.

You can adjust the pitch of the noise slightly (± 400 cents, or 4 semitones), either to simulate PAL 50Hz noise or to tune it along with your music.

Some units generate different noises depending on external factors, for instance, it is well known that the Commodore 64 and Vectrex background noises were directly related to what was displayed at a particular time on the screen. In this case various noises are available.

A quick look at the available selection:

- Atari 2600
- Atari STf
- Atari 800XL
- Bally Astrocade
- C64’s (brown breadbox)
- C64C with 8580,
- C64 (through the RF)
- C128
- GameBoy (DMG-1)
- GameBoyColor (CGB)
- Gemini (Atari 2600 clone)
- Genesis 1 (using headphones – two levels), using the audio out, and the RF modulator
- Intellivision I (RF)
- Intellivision III (RF)
- Leisure Vision
- NES and VS arcade PCB
- Odyssey2
- Casio PV1000 RF (good luck finding your own :)
- Sega Master System (RCA and RF)
- Super PacMan arcade PCB
- Vectrex (4 different display patterns)
- VIC-20
Cabinet

A very efficient zero latency(*3) convolution engine allows chipcrusher's sound to go through a few of the most well loved vintage gaming devices, monitors and enclosures. This helps to control the harshness in the sound that can be introduce by the DAC encoding. Each impulse gives a totally different tonal quality to the sound. You can dial in the unaffected sound as well for added flexibility.

(*3)Note all impulse responses were trimmed with the maximum amplitude at 32 samples from the start of the file at 96kHz. You can consider this little delay to be the 'response time' of the speakers.

We have split the available impulses into the following categories:

Computing

Speaker and casings impulses of various PC clones but also three Apple classics, the Apple IIe and the Apple IIGS.
Filters

Mostly filled with the 3 variants of AMIGA Original Chip Set machines (A500, A1000 and A2000), with the infamous LED filters on and off. We also included the extreme lowpass filter from the SK1.

Monitors

From the classic Commodore monitors used with the C64, to the Mac Plus and even a 50s Television set.
Most portable gaming devices, or speaker-bearing devices of the day including a Coleco Mini-Arcade, and a fullsize Galaxian Arcade Cabinet and Nintendo VS cabinet. A Oui-Mote, a Vectrex, 4 different GameBoys (DMG-1, CGB, GBA and GBA clam shell) and more!
Various classic drum machines, keyboards which came with a speaker.
**IMPORTANT NOTE ON PRESETS:**

**.fermatax files:** full content of the UI is saved in xml form, including values of parameters's at the time of saving. (VST hosts fxb/fxp files are similar in that they contain the same data)

**.fermatap files:** presets to sub portions of the UI, - in chipcrusher 2, this basically means the SPC Delay.

---

**Latency**

Since the DAC Emulator juggles between 3 distinct sample rates, introducing latency is sadly inevitable.

Upon a session start, chipcrusher will warn the host (VST/AU only) about a worst case latency for the current host sample rate setting, so that changing the internal 'virtual' sample rates does not de-synchronize the outputs with regards to the other tracks.

However, since the DAC Encoding section can be bypassed, we have added internal delay as well to compensate for the possibility of changing the section to bypass while the host plays.

Since PDC (plugin delay compensation) behavior varies greatly from one host to the other, your tracks might need to be rendered and offset for the best results.

Note: Turning Live Mode On in the settings tab will force reported host latency to be nil AND internal delay as well. This is useful not only for live situations, but also if you only use the Cabinet section.
The Settings Page

You can access the Settings section of chipcrusher by clicking on the top section:

This will show you the credits (including version number) and some vital info on the host’s audio engine settings. This info can be useful if you ever need technical support.

**Live Mode:** This button disables latency compensation data sent to the host program as well as some internal overhead. That way, chipcrusher can be used in a live setting as an effect without causing additional latency (More on that in the **Latency** section).

**UI Zoom:** Increase user interface size. Use this if you have a high DPI monitor.

**Bypass DAC %:** Level of bypassed signal mixed into DAC Encoding output.

**PWM/PDM Carrier:** Enables carrier output on PWM and PDM encodings.

---

**Warning!**

The carrier is a piercing high tone at screamingly loud volumes.

Clicking on the **License** button will display your license Keycard. Click again anywhere on the interface to close the image. You can check for updates to the Fermata engine and for chipcrusher using the **Update Engine** and **Update Plug-in** buttons.

To go back to the main controls, click on the X or outside the pop-up box.
**Getting Help**

The first place to look for a solution to any problem you may be experiencing is in this manual. Please read the manual before contacting support. Next, check the readme files (if any) which contain important information and all last-minute changes that haven't been available when creating this guide.

Plogue chipcrusher is evolving and growing. Please check the support area of our website at www.plogue.com for the latest up-to-date information products, troubleshooting, FAQs, helpful hints and tutorials. Another resource is the support forums.

Whenever you encounter problems, you should also check if you have installed the latest updates. The version number of your software is displayed in the Settings dialog. (click on center-top of the interface), in the credits section on the top left. Updates are released regularly to fix known problems and to improve the software.

If you can't find a solution to your problem please email us at chipcrusher.support@plogue.com. The best way to get the help you need is by giving us plenty of detailed information about the problem you are having. We do ask you to read this guide thoroughly and exhaust the other avenues of support before contacting us.

The Plogue forum can be accessed at: [http://www.plogue.com/plgfrms3/](http://www.plogue.com/plgfrms3/)

You don't have to register to browse posts, but before you can post, you will have to sign up.
Acknowledgments

Producing Plogue chipcrusher would not have been possible without the combined help, talent and support of many extraordinary people. I am grateful to those who have contributed and would like to thank them all.

Jeff Hurchalla: for your incredible zero latency and efficient convolver and deconvolver utilities which were of great help.

Previous research (no particular order):
Lord Nightmare, balrog, kevtris, Blargg, Antti Lankila, Brad Taylor, Sebastian Tomczak, and the MAME team.

Testing and suggestions (no particular order):
A_Rival, XC3N, 8GB, Chupathingy, 4Mat, sink, Peter Swimm, Bill Hicks, Bill Prokopow, Brady Leo, Chad Beckwith, polymur, tssf, Dsmolken, Johnny Collazo.

Last but far from least:

Martine for enduring the hoarding of all that “crap” in the basement, and my long technical gibberish rants.
And my two wonderful sons, for sometimes being a d___ when my c___ doesn’t work.

David Viens, November 2018