INSIDIOUS

by Mike Clarke

Table Of Contents

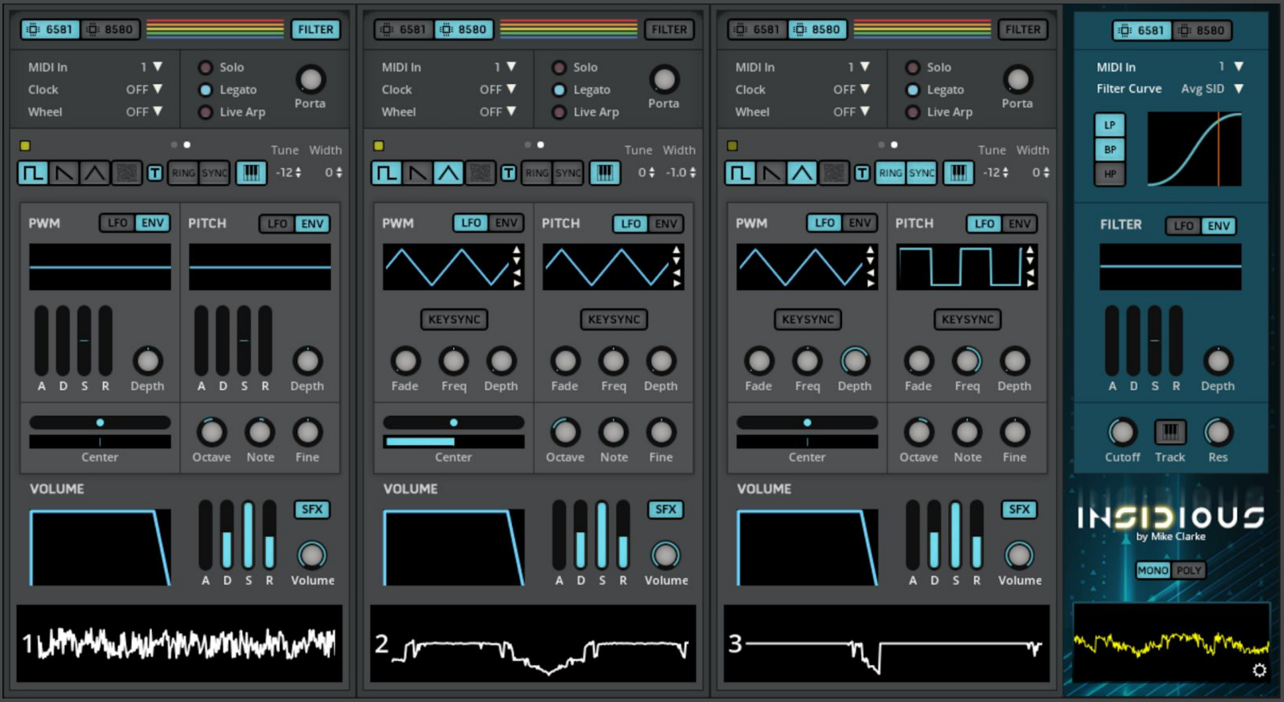
Introduction **3**
- About the SID 3
- SID Features 3
- 6581/8580 Differences 4
- The SID Sound 4

Synthesis **6**
- Root Waveforms 6
- Pulse Width Modulation 6
- Combined waves 6
- Ring and Sync 7
  - Ring 7
  - Sync 7
- Step Table 7
- TEST bit 8
- Clock 8
- Filter 9

User Interface **10**
- General usage 10
- Mono/Poly Mode 10
- Chip and filter selection 10
- System controls 11
  - MIDI In 11
  - Clock 11
  - Wheel 11
  - Pitch Bend Range 11
- Play Controls 12
  - Solo 12
  - Legato 12
  - Live Arp 12
- Wave Selector 12
  - Root Waveforms 12
  - TEST Bit 12
  - Ring Modulation and Hard Sync 13
  - Key Tracking 13
  - Coarse Tune 13
  - Width 13
- Step Table 13
  - Speed 14
  - Loop Start and Loop End 14
- Modulation Controls 14
  - PWM 14
  - Pitch 14
  - Volume 15
  - Common Modulation Panel 15
    - LFO 15
    - Envelope 16

SFX Button 16
Oscilloscope Display / Mute 16
Filter 16
Chip selection 16
MIDI In 16
Filter Curve 17
Modulation Controls 17
Cutoff and Resonance 17
External Filter Input 18
Options 18
Enable MIDI velocity 18
Enable scopes 18
Remove circuit noise 18
Global Tuning 18
For Owners Of The Full Version Of Reaktor 18

Appendices 19
A: inSIDious / Real Hardware Differences 19
6581 19
8580 19
Both 19
B: Control Change Assignments 20
C: Volume Envelope Values 20
D: Further Reading 21
E: About the contributors 22
Chris Huelsbeck 22
Fabian Del Priore 22
Jason Page 22
LMan 22
Jammer 23
TDK 23
Martin Galway 23
Rob Hubbard 23
F: Thanks to 24



Introduction

Thank you for purchasing inSIDious, the best SID emulation available for your DAW, and welcome to the wonderful world of the MOS Sound Interface Device.

inSIDious has been developed over 5 years by Mike Clarke, a veteran video game musician and programmer, who spent his formative years playing Commodore 64 games and listening to their music. It was this music that set him on the path that defined his entire career and gave him huge respect for and admiration of the SID chip and the sounds that it could produce. So much so that he used its sounds in many and varied music productions over the years. However, he felt that none of the existing SID VST attempts were anywhere near suitable as a replacement for his HardSID Quattro sound card (containing 4 actual SID chips), which was reaching the end of its usable life. So he took it upon himself to rectify the problem by creating his own emulation to be as close as possible to the real thing. As a result, inSIDious bears little resemblance to the many SID-like VSTs floating around that have versions of the basic waveforms and little else. It is a reimplementation of every feature of the chip as accurately as possible. Apparently, Mike also writes manuals in the third-person.

About the SID

The SID chip was created in 1981 by Bob Yannes of MOS Technology who felt that existing computer sound chips of the time were unsatisfactory. He set out to create something more akin to the prevailing subtractive synthesizers and in only 5 months Yannes and his three technicians created something truly extraordinary: A combined analog/digital synthesizer utilizing three digital oscillators and an analog filter. As amazing as the achievement was, the finished product never lived up to Yannes' hopes. In fact, he later went on to found synthesizer company Ensoniq, and stated that the Ensoniq ESQ-1 was the synthesizer that he intended the SID chip to be. Nevertheless the SID chip was so far ahead of the competition in the computer and video games space that nothing came close to its sonic capabilities for the rest of the decade.

There are 2 major versions of the SID chip, numbered the 6581 and the 8580 (aka 6582). On paper, both have the same feature set, however, the 6581 contained a number of flaws in its design and was subject to affectations in the manufacturing process. These errors contribute to the sound generation in a number of ways that make the 6581 truly unique. The 8580 was designed to fix these errors and as a result has a sound that is more accurate to the original specification or, as some would say, is more boring. While both are broadly the same, they are also subtly different.

SID Features

The SID chip has three independently accessible digital oscillators that can generate 4 fundamental waveforms: Pulse, Sawtooth, Triangle, and Noise.

The Pulse waveform can have its pulse width set to any one of 4096 positions that represent the complete range from a pure square wave to a pulse thin enough to be silent.

The Noise waveform is a digital pseudo-random noise generator that can be pitched up and down and sounds characteristically "8-bit".

The Pulse, Sawtooth, and Triangle waves can be enabled in combination in any configuration, giving rise to a unique set of extra waveforms.

Any oscillator can be synchronized to the frequency another to produce an effect known as Hard Sync.

When an oscillator is set to the Triangle waveform (or Triangle combined with Pulse) it can use one of the other oscillators as a carrier frequency to produce ring modulation, which can produce bell-like metallic sounds and effects.

Each oscillator has its own envelope generator, which allows an ADSR envelope to be applied to the volume.

The filter is an analog multimode resonant filter that can be set to low-pass, band-pass, high-pass, or any combination of the three. An audio input can route external audio through the filter.

6581/8580 Differences

There are a few major differences between the 6581 and 8580 that are of course emulated by inSIDious:

- The combined Pulse/Sawtooth/Triangle waveforms are different.
- The 6581 waveforms are louder than the 8580 and have some extra unintended harmonics.
- Due to poor tolerances in the manufacturing process, the filter on every 6581 has a different cutoff curve.
- The 6581 filter has a bug that causes the input level to affect the output signal and cutoff frequency in a non-linear way. This makes the 6581 filter sound truly unique with a warm characteristic and ability to add a pleasing distortion to the signal.
- The 8580 filter, being closer to the intended specification, has a higher resonant peak.
- The 6581 filter has a lower output volume.

The SID Sound

The SID is lauded for its unique sound, but what makes it so? On paper, it's really nothing special. It did not present any new type of synthesis, nor did it present its features in a particularly clean way. It's a noisy, buggy, weird, raw sound chip. But on the positive side, it's a noisy, buggy, weird, raw sound chip. It's got a lot of character. But it's not just the features of the chip that give it its distinctive sound, it's how those features were used in the context of video games in the 80s.

Early SID musicians used the SID in its most basic form, with very basic usage of simple waveforms to play simple melodies. Mostly these were programmers who dabbled in music or who just programmed some rudimentary music out of pure necessity. But later, the likes of Rob Hubbard, David Dunn, and Martin Galway came along. Rather than being programmers who dabbled in music, these were first and foremost excellent musicians who also happened to be great programmers. They were not satisfied with the simplistic results of just playing melodies using simple tones and began to experiment with ways to make their music sound more rich and complex. This involved programming rapid pitch and waveform changes, and making heavy use of the pulse waveform with sweeps of its pulse width parameter. Some of these techniques, which have come to define a large part of the modern chipmusic genre, would later make their way onto other platforms such as the NES, Atari ST, and Gameboy, as those original musicians or their peers branched out from the Commodore 64 and the techniques spread.

At the beginning of this new SID era of music pioneers, long before the 8580 came along, the filter was an exciting prospect to enable the creation of interesting timbres. David Dunn's early works were especially filter-heavy, because why wouldn't anyone use such a useful feature? Unfortunately, the broken nature of the 6581 wasn't understood very well at the time. As mentioned above in the 6581/8580 differences, the 6581 suffered from a problem during manufacture whereby different batches of the chip could end up with drastically different filter cutoff curves. On one chip, a value that would set the low-pass filter to cutoff at a pretty low frequency might have little to no effect at all on a different chip. Sometimes, it would be so bad that on some machines what was supposed to be a mildly filtered sound would be filtered to almost complete silence. This could completely destroy the composer's intent and many, if not most, people would hear a different sound to that intended. As a result, in a relatively short period of time, usage of the filter almost stopped completely, with only a handful of tracks using it in a few choice places. And what a shame that was because the 6581 filter is unlike any other with its warm distortion and generous helping of craziness. Nevertheless, even without the filter the SID was still way ahead of everything else in terms of sonic capability, and some filterless works of genius still came to fruition on the SID.

Since then, the SID scene has gone from strength to strength and modern SID composers almost exclusively target the 8580 due to its stability: What is written using one 8580 chip is pretty much guaranteed to sound exactly the same on all 8580 chips. Some of the recent compositions from musicians such as Linus and Jammer are true masterpieces. Such is their technical prowess that with many of their tracks it is difficult to believe that such sounds are coming from the SID.

Synthesis

Root Waveforms

The SID has 4 fundamental waveforms, Pulse, Sawtooth, Triangle, and Noise, which can be affected by volume, pitch, hard sync and by combining them together in various combinations. The Triangle wave can also be affected by ring modulation, while the Pulse wave can have its Pulse Width set to any custom value.

Pulse Width Modulation

PWM is one of the SID chip's most instantly recognizable sounds. In fact, many times have other soundchips been described as performing "SID emulation" solely because someone managed to squeeze a PWM sound out of it. However, it is by no means unique to the SID.

A pulse wave consists of a waveform that switches between maximum and minimum at a regular interval. We call the time spent at maximum the width, and by adjusting (modulating) this width we can change the tone of the sound. The most common usage of PWM on the SID is to sweep the width from one value to another and back again, giving the classic "neeeooowwww" sound.

In scientific and electronics terms, the width value is called the 'duty cycle' and is usually expressed as a percentage. At 50%, the pulse waveform is a perfect square wave, while the extremes of 0% and 100% produce complete silence. Less capable sound chips such as those on the NES and Gameboy only allow pulse duty cycles of 12.5%, 25% and 50%, whereas the SID allows the complete range from 0% to 100% in 4096 steps.

inSIDious uses values from -1.0 (representing a 0% duty cycle) to +1.0 (representing a 100% duty cycle). A value of 0.0 gives a perfect square wave.

For the classic sweeping PWM effect, set the Center control to 0.0, set the LFO depth to 1.0 and the LFO frequency to your preference.

Combined waves

The waveform generator in the SID is digital and this gives the ability to combine the root waveforms in an unusual numeric way. The original design was supposed to logically AND together the waveform values, but this isn't actually the case. In fact, the mechanism behind how the waveforms are combined was a mystery for many years due to its esoteric nature. A hardware bug actually makes these waveforms subtly different on every 6581 chip, although the 8580 always produces identical waves between chips.

On the 6581, combining the Triangle with the Pulse or Sawtooth waves gives a distinctive metallic, scratchy type of waveform. Other combinations give waves that are not very useful as they are so thin and quiet, but they have their place now and then. On the 8580, all of the combined waveforms are useful, although most sound pretty similar to each other.

When a Pulse wave is one of the combined waves, the waveform is affected by the Pulse Width, silencing part of the wave relative to the width value. This can sometimes be a little confusing as you may end up with a silent waveform without realising why. If this is the case, the PWM indicator will turn red to let you know. You can use this with the PWM LFO to create a different type of sweeping PWM sound.

It's worth noting that when the Triangle and Pulse wave are combined, the resulting wave can still be affected by Ring Modulation (see below).

The Noise waveform can not be used in combination with the other waves. Although it is possible to do so on a real SID, it causes the noise generator to lock up and produce silence.

Ring and Sync

Ring Modulation and Hard Sync both need two channels to work. These effects require two signals known as the Carrier wave and the Modulator wave. The Carrier wave is the root waveform that defines the fundamental shape and pitch of the waveform. The Modulator wave affects the Carrier wave in interesting ways to produce new harmonics.

On the SID, the setup is a little confusing. When enabling Ring or Sync on a channel, it becomes the Modulator wave and the previous channel its Carrier wave. "Previous channel" means that channel 3 is affected by channel 2, channel 2 affected by channel 1, and channel 1 affected by channel 3. It is okay to turn off the "previous channel" by switching off all of its waves, reducing its volume to 0 or muting the channel. The ring or sync will not be affected.

Ring

Ring Modulation is a type of Amplitude Modulation that multiplies one waveform by another. It produces a sound that has an eerie bell-like quality to it. Usually, the Carrier wave is directly multiplied by the Modulator wave, but on the SID this is not quite the case. Firstly, Ring Modulation only works on a channel that is using the triangle wave. Secondly, the pitch of the previous channel is used to generate a square wave that is multiplied with the Modulator.

For an example of this effect, set the pitch LFO of one channel to have a depth of 24. Then in the next channel, use the triangle wave and switch on the Ring button. Or instead swap them around and put the LFO on the channel that is using the Ring effect, which will give a different sound.

Ring modulation can be used for subtle modulation or for crazy effects like lasers and even a human-sounding voice if you get the settings right.

For instant results, deselect all waves in channel 1, then in channel 2, select the triangle wave and switch on Ring. Now hold down a note and in channel 2 drag the Note value up and down.

Sync

Hard Sync is actually a very simple effect that produces a great sound. Basically, the Carrier wave is restarted at the frequency of the Modulator wave. The best way to set this up is to use a pitch envelope on the Modulator channel.

To try it out, follow the instructions in the Ring description above, but switch on Sync instead of Ring (or you can switch on both). Note that where Ring Mod has to use the triangle wave, Sync will work with any waveform.

Step Table

Because of the limited number of channels, it was a constant battle to make the SID sound like it was playing more than 3 channels at once. One way that helped to do this was to change the waveforms quickly at the start of each note. This technique can give a harsher attack to sounds, can change the perceived tone of sounds, and can also be used to emulate percussion sounds and create sound effects. For certain situations, it can also make one channel sound like it's playing two instruments. For example, placing a short noise click at the beginning of a bass sound and playing the sound on every quarter note can make it sound like it's playing both a hi-hat and a bass at the same time. In inSIDious, the Step Table enables all of these things by giving you access to a loopable step sequencer where you can change the waveforms, pitch, and pulse width values. You can even synchronize it to your DAW.

TEST bit

On every synthesizer, the oscillators are usually running in the background continuously. If there is silence, the oscillators will still be playing, but you won't be able to hear them because the amplitude envelope has reduced their volume to zero. When you press a key to play a note, the envelope allows the sound to pass through, but the oscillator itself does not restart its wave from the beginning. The test bit allows you to do just that. When it is set, the oscillator will have its wave restarted from the beginning (at a phase of zero). This is an advanced feature that is mostly useful when using multiple channels to create a single monophonic sound. If you set one channel to use the test bit and another to not, the two waves will have a random phase difference on every new note. You can also set the test bit at any point in the step table, allowing you to make it sound like a single waveform is playing a new note.

Try using 2 channels, both set to use the sawtooth wave and make sure the TEST bit is disabled for both. Play some notes to hear how it sounds. The sound should be consistent on each new note, but the phase between the 2 channels will be different. You can see this on the output scope at the bottom-right of inSIDious. Now enable the TEST bit on one channel. Play some notes again and you'll be able to hear (and see in the scope) that the phase is different between the two on every note. This is because one channel has its phase reset on every note while the other doesn't. Now enable the TEST bit on both channels. Because both channels have their phase reset on every note, the two waveforms are identical and so sound like a single channel at twice the volume.

Clock

It is not widely understood among the general populace how early home computer and console-based game music actually worked, so here's a quick overview.

All of the early machines were designed to be used in conjunction with a standard television from the era. This fact forced a very specific limitation onto every game: For smooth play, every update to the graphics must be performed in synchronisation with the update speed of the television display. In NTSC regions (USA, Japan, South Korea, some South American and Asian countries) this speed was 60Hz (60 times per-second). In PAL/SECAM regions (the rest of the world) this speed was 50Hz. It should also be noted that handheld consoles deliberately followed this system and all have a screen update of 60Hz.

Because of this, each game would have to perform all of its calculations and move the graphic positions at a very regular interval before the next screen refresh. In game programming terms, this repeating cycle of operations is called the "game loop" and at the highest level can be as simple as "Read controllers. Update game logic. Update graphics. Wait for screen refresh. Repeat.". As this is the top level of control of all game operations and occurs on a strict time interval, it makes sense to add in an extra step into the game loop to update the sound. And so, the standard game loop that pretty much every game would adhere to (and which games effectively still do today) would be something like: "Read controllers. Update game logic. Update graphics. Update sound. Wait for screen refresh. Repeat.".

In terms of game sound it means that apart from a few special cases where extra code is used, the parameters of the sound chip could only be changed every 50th (or 60th) of a second. Any change of wave, any change in pitch, any change of PWM value, has to be on an exact time step of 50Hz or 60Hz.

In practice, there are only a few instances where this is an audible effect. One is when switching quickly between notes to create the classic 8-bit arpeggio sound. For any emulation to sound realistically 8-bit, the arpeggiation speed *has to* be at 50Hz or 60Hz or a division thereof. In terms of standard synthesizers, that translates to an arpeggiation speed of 20ms or 16.6666ms respectively.

Another noticeable effect is when doing fast pitch slides or modulation. Instead of a smooth slide, you can hear steps as the pitch jumps from one value to the next. You can hear this very clearly in many 8-bit game sound effects such as lasers.

One other effect is when switching waveforms as quickly as possible. You can create many new sounds (especially percussive sounds) by switching waveforms very quickly. The faster you can switch, the more possibilities are open to you. The restricted switching speed contributes to the 8-bit-style sound by keeping such sounds a little more primitive and rough.

You can switch on the Clock in inSIDious to restrict parameter updates to the selected frequency, thus emulating this scenario.

Filter

The basis of subtractive synthesis is the filter and the SID filter is (at least on paper) a pretty standard implementation of a multimode filter. One interesting aspect is that the filter modes are not exclusive, meaning that you can enable low-pass and band-pass at the same time, or any other combination of the three modes.

The 8580 filter, for all intents and purposes, behaves exactly how one would expect a standard mathematically correct filter to behave. It sounds pretty much like any other standard analog filter that you might have heard on any other synthesizer.

The 6581 filter, however, is rather a different beast due to bugs present in the hardware. It does not conform to its intended specification and so has a much more interesting character. Firstly, the tone is different to the 8580, giving it a bit more of a weighty sound. Secondly, it introduces a very unique type of distortion when the input signal is too loud. The distortion is commonly described as being "warm" and results in a very pleasing tone that is truly unique to the chip. It is most noticeable when sending multiple channels through the filter as doing so boosts the input signal beyond the distortion threshold. For an instant result, set the filter to low-pass (LP) and set the cutoff to a low value. Then send 2 channels through the filter and set the Note pitch value of one of the channels to -5. If you then switch the filter between the 8580 and 6581 modes, you'll really hear the difference.

User Interface

General usage

inSIDious has been designed to give access to every feature of the SID in the simplest, most musical way possible. It also has features that are not strictly properties of the chip per se, but of the way it was programmed using the processor of the Commodore 64. This allows you to easily make a huge range of sounds from simple waveforms to sound effects to chip drums to massive mono leads and basses. It can also be used to make recreations of any actual Commodore 64 music.

All knobs have a dual name/value display. When you place your mouse over one of the knobs, its name will be replaced by its current value instead. The ADSR slider shows their value to the upper-right of the sliders. If you right-click on a knob or slider it will return to its default value, which is zero for most controls.

Any knobs that set a frequency value can be synchronized to the tempo of your DAW. Turning such a knob to the right will set an absolute time in Hz (cycles per-second). Turning to the left will allow you to set a note value that represents a frequency that is synchronised to your DAW. You can set the note value from 4/1 for 4 whole notes down to an 1/8th note (or 1/16th note for the Arp Spd and Table Speed controls). Values that have a T at the end (such as 1/2T) are a triplet equivalent of the specified note.

All controls have in-built popup help that can be enabled by switching on the ⓘ Info Hints button at the top of the Reaktor Player window.

Mono/Poly Mode

MONO POLY

As this button is tucked away from the rest of the controls, it's best to address it first. Down in the bottom-right corner of inSIDious is the Mono/Poly button.

In Mono mode, each of the three channels will act as a single-oscillator monophonic synthesizer. If they are all controlled collectively by all being set to the same MIDI channel, then inSIDious will act like a three-oscillator monophonic synth.

In Poly mode, inSIDious will act like a single-oscillator synthesizer with three notes of polyphony. Unlike ordinary synthesizers of this type, each note of polyphony corresponds to one of the SID channels. This means that each note can have a different sound and this can produce some very interesting and unusual effects.

Chip and filter selection

6581 8580

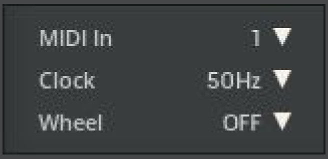
At the top-left of each channel is the chip selection button used to switch between the 6581 chip and the 8580 chip. In 8580 mode, you will have slightly quieter waveforms and different combined waves to the 6581. On the 6581, the Sawtooth + Triangle and Pulse + Sawtooth + Triangle waves are very quiet and almost useless. On the 8580, all of the combined waves are usable.

FILTER

At the top-right is the filter button. Switching this on will route the channel through the analog filter.

System controls

Beneath the chip selection button is the system panel. There are three menus:



MIDI In 1 ▼
Clock 50Hz ▼
Wheel OFF ▼

MIDI In

Here you can set the MIDI channel used to control the SID channel. Each SID channel can be accessed independently as required. With all SID channels set to the same MIDI channel, inSIDious will effectively operate as a monophonic 3-oscillator synthesizer. However, using each SID channel independently by having each on a different MIDI channel will generally give a much stronger retro 8-bit-style sound due to the more simplistic tones.

Clock

Game consoles almost always updated their audio parameters only whenever the video screen was refreshed. In NTSC territories such as North and Central America and Japan this was at 60Hz (60 times per second), and in PAL and SECAM territories (as in most of the rest of the world), this was at 50Hz. When the Clock is enabled, all parameter updates will be restricted so that they only update at the frequency of the clock. This gives a subtly more 8-bit sound, especially when using fast pitch changes, and is definitely recommended to be enabled if you are going for a dirty retro sound. See the Clock section in the Synthesis chapter for more details.

The Clock also controls the maximum frequency of the step speed of the Step Table (detailed later in the document) and can be set to 100Hz, 150Hz, or 200Hz. In modern SID usage, these are known as 2x, 3x, and 4x speed and such speeds can produce interesting advanced effects.

Wheel

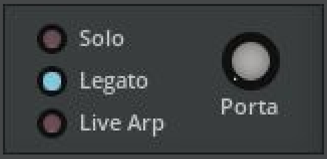
Using this control you can set the desired destination of the MIDI modulation wheel. Available options are:

- **OFF** The modulation wheel will do nothing.
- **PWM Center** Controls the Center control of the Pulse Width Modulation.
- **PWM LFO** The modulation wheel will control the Depth knob of the PWM LFO section.
- **PITCH LFO** The modulation wheel will control the Depth knob of the PITCH LFO section, which is used for vibrato. In Wheel mode the maximum LFO range is restricted to 7 semitones.

Pitch Bend Range

The default range for MIDI pitch bend is 2 semitones. This can be changed by revealing the "B" panel. RIght-click on the coloured bars at the top of the SID channel and choose "View B". The system panel will then switch to show the Pitch Bend Range control. To switch back again, right-click again on the coloured bars and choose "View A".

Play Controls



The Play Controls relate to how inSIDious responds to the incoming notes.

There are three play modes:

Solo

Retriggers the Step Table and volume envelope on every note.

Legato

Overlapping notes will jump to the new note without retriggering anything. Releasing the overlapping note will return to the original note if it is still being held down.

Live Arp

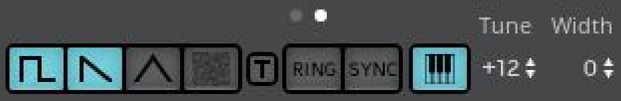
Live Arp operates like Legato mode, but the pitch will jump quickly between each held-down-note at the speed specified by the Arp Spd control. At speeds of 50Hz or 60Hz, this gives the classic bubbly 8-bit chip music effect.

When Solo or Legato is selected, the control knob shows the Portamento control. Setting this to any value above 0.0 will cause overlapped notes to slide from one to the next. It will also cause a slide between any pitch values that have been set in the Step Table.

When Live Arp has been selected, the control knob shows the Arp Spd control. Turn it to the right to set the arpeggiation speed in hertz or to the left to synchronize the speed with the tempo of your DAW.

Wave Selector

The Wave Selector is where you can set the root waveform for the channel and set a few other properties. It is actually the first row of the full Step Table, which you can read about below. You can switch between viewing the full Step Table or the Modulation Controls panel by clicking on the two dots above the selector buttons.



Root Waveforms

The four buttons from the left are the waveform buttons representing Pulse, Sawtooth, Triangle, and Noise. If none are selected, no sound will be produced by this channel other than some circuit noise when you send a Note On or Note Off. The Pulse, Sawtooth, and Triangle waveforms can be enabled in tandem with each other to enable one of the SID's combined waveforms. The selected waveforms are mathematically combined together giving some unusual sounds.

If the Noise waveform is selected, the other buttons are automatically deactivated as the Noise cannot be combined with any other (on the SID chip, combining noise with anything else locks up the noise generator and produces silence until the channel has been reset).

TEST Bit

The small T button represents utilization of the SID's TEST bit. When this button is enabled, the waveform will have its phase reset to 0 on every note. It is a very subtle effect, but can be useful for certain sounds. See the TEST Bit entry in the Synthesis section to create an example of its effect.

Ring Modulation and Hard Sync

Next we have the RING and SYNC buttons. The SID can link 2 of its channels to produce a hard sync effect or ring modulation. When either of these are set, the previous channel is used for the modulator frequency while this channel is used as the carrier wave. Hard sync works with any of the non-noise waveforms while ring modulation works only with the Triangle and combined Pulse + Triangle waveforms. These are explained in more detail in the Synthesis section.

Key Tracking

When the button is enabled the pitch of the channel will follow the incoming MIDI note and will respond to pitch modulation.

When set to half mode the channel will use middle C as its root note, but can still be pitched by pitch modulation controls.

When disabled completely, the pitch will ignore both the incoming MIDI note and pitch modulation, but can still be pitched using the Coarse Tune control in the table or the Octave, Note, and Fine controls in the channel. This feature is mostly useful when used as part of a larger Step Table sequence, or when using the channel as a modulator or carrier wave for when ring modulation or hard sync is used.

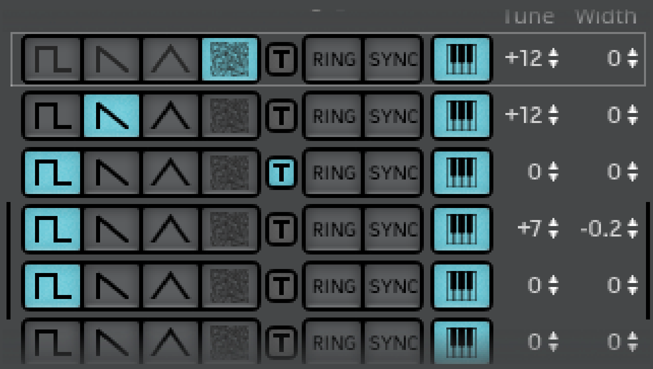
Coarse Tune

The Tune control allows the channel to be pitched up or down. This offers similar functionality as the Note control in the Pitch Modulation section, but allows a pitch change at each step of the Step Table. The value can be adjusted like any knob control; hold down the mouse button over the value and drag up or down.

Width

The Width parameter adjusts the Pulse Width value in steps of 10%, and works in addition to the PWM controls in the Modulation Controls panel. Again, this is mostly useful when used as part of a larger sequence of the Step Table.

Step Table

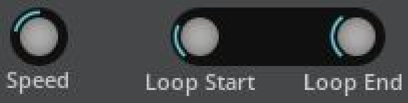


The Step Table is a way to use inSIDious as a simple wavetable synthesizer in the same way that the original 8-bit musicians did. It is modelled on how music drivers were programmed on the Commodore 64 and can be used to create various sound effects, 8-bit drums, and other interesting and unusual sounds.

There are a maximum of 16 steps and inSIDious can step down the list at the specified speed each time it receives a new note.

Each row of the Step Table consists of one complete Wave Selector entry. When a step is activated, the settings of that row will then be used. This allows you to change the waveform, pitch, and PWM width over time within the sound.

Speed



Beneath the 16 rows are three control knobs. The first, labelled Speed is used to control the speed at which each row is selected. As with other frequency controls, turning it to the right will allow you to set a frequency in Hertz, turning it to the left will allow you to choose a note quantity for DAW synchronisation.

The speed is ordinarily limited to 50Hz, but this limit can be increased to up to 200Hz by setting the Clock to the appropriate value.

Loop Start and Loop End

The other two knobs are Loop Start and Loop End. These are used to set the loop points of the Step Table. When these are non-zero, a grey box will surround one of the rows to show you the current step position. Also, a black vertical line will appear on the left and right edges. These lines show the current start and end points of the loop.

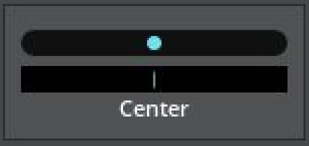
When you play a note, the position marker will step down the rows from the top at the speed specified, adjusting the playing sound to match the row settings. When it reaches the Loop End position, it will jump to the Loop Start position on the next step. If Loop Start and Loop End are equivalent, the marker will stop and hold at that position.

When you switch to the Modulation Controls panel when the Step Table is active, a light will be visible to the upper-left showing it in operation.

Modulation Controls

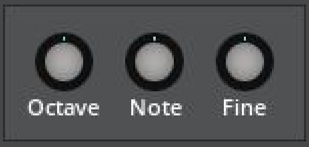
The Modulation Controls panel is split into three main sections, PWM, Pitch, and Volume. The PWM and Pitch panels both use a common modulation panel that is also used in the Filter panel. The common panel is detailed after the non-common controls.

PWM



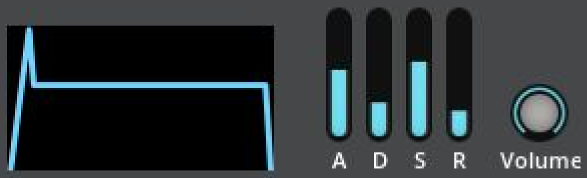
For details on what exactly PWM is, see the Synthesis section. This control sets the root value of the Pulse Width Modulation. Dragging the knob left or right will change the value and thus the pulse width when a pulse waveform is selected. Beneath the knob is the value display. This shows the current PWM value, which consists of the Center control plus the PWM LFO value plus the PWM Envelope value plus the Wave Selector's Width value. The PWM value is usually active from -1.0 to +1.0 (some combined waveforms operate from -1.0 to 0.0), but due to the summing of all of the relevant parameters, the value may go beyond this range and result in silence. If this occurs, the display will show a red marker in one half or the display, corresponding to which boundary the value exceeded.

Pitch



These three controls allow you to set the root pitch of the sound. Octave will shift the pitch up in octaves, Note will pitch up or down by up to 48 semi-tones. Fine will shift the pitch up or down within the range of a semitone.

Volume

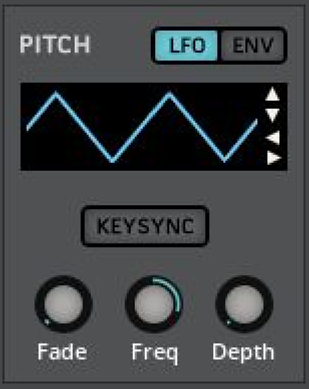


Here we have the volume ADSR. It operates just like any other ADSR envelope that you might have seen. Moving the sliders will update the envelope graph so you can visually get an idea of what it will sound like. The slider values match those of the SID and go from 0 to 15. Each value represents a specific time and you can see those times by holding your mouse over the relevant control when Info Hints is enabled or by checking the table at Appendix C in this manual. The Attack phase is linear and the Decay and Release phases follow a curve that matches the SID.

Common Modulation Panel

This panel contains the LFO and Envelope controls that relate to either the PWM, Pitch or Filter, depending on which section it is contained in. The panel can be switched between the LFO controls and the envelope controls using the LFO ENV button.

LFO



The LFO panel contains a large display showing you the currently selected waveform along with some controls to modify the waveform itself. To the right of the waveform display are four triangle-shaped buttons. These are used to change the type and phase of the waveform. By using the up or down buttons, you can cycle through triangle, sine, square, ramp up, ramp down, and random waveforms. The left and right buttons allow you to choose the phase of the waveform. The phase dictates the point at which the waveform starts and is mostly useful when KeySync is switched on. The available phases are 0°, 90°, 180°, and 270°, and the display will show the waveform with its starting point at the selected phase.

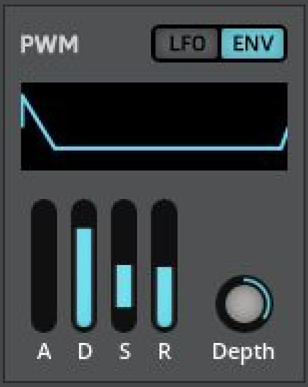
If the KeySync button is enabled the LFO will restart from its starting phase point whenever a note is played.

The Fade knob allows you to slowly ramp up to the LFO's selected depth over a period of time. The value is in seconds.

The Freq knob determines the speed of the LFO in seconds. At the frequency specified, the LFO will complete one cycle of the waveform.

The Depth value sets how intense the LFO will be. This value can be different depending on which LFO it pertains to. For PWM, it goes from 0.0 to 1.0. The LFO is bipolar and cycles between positive and negative, so a value of 1.0 means that the PWM will cycle from -1.0 to +1.0 to cover its full range. For Pitch, the Depth value goes from 0.0 to 24.0 representing a maximum range of -24 to +24 semitones. For the Filter, the range is 0.0 to 1.0 and applies to the cutoff frequency position.

Envelope



The modulation envelope is a bipolar envelope (it can go both positive and negative), so is probably a little different from what you're used to. As you change the parameters, the large display will change in realtime to give you a visual representation of the envelope shape. The Attack, Decay, and Release sliders are basically the same as any other normal linear ADSR; they allow values from 0.0 to 8.0 seconds. The Sustain value is unusual in that you can use it to set a sustain level between -1.0 and +1.0. This expands the capability of the envelope and allows you to perform a full sweep across the full positive and negative range. Also, the Depth parameter, which sets the intensity of the envelope, can be set to a negative value, which allows you to flip the whole envelope upside down for a greater range of control. Finally, setting the Release value up to its maximum will switch it to become infinity and the value display will show an infinity symbol (∞). The envelope will then hold itself at its last value when you release a note.

SFX Button

The SFX button is the magic 8-bit sound effect button. Pressing this will randomize the Step Table and the modulation controls of the related channel to produce an instant crazy sound.

Oscilloscope Display / Mute

At the very bottom of the panel is the oscilloscope display. This shows the current raw waveform being output by the channel. To its left is a number showing the SID channel number. Clicking on this number will allow you to mute or unmute the channel. A red X will appear over the number when the channel is muted. Note that even when the channel is muted, it can still be used to modulate the next channel if the next channel is using Ring or Sync (see the Synthesis section for details of those features).

There is a fourth oscilloscope display at the bottom-right of inSIDious. This is the output scope and shows the final audio output after channels have been mixed and optionally sent through the filter.

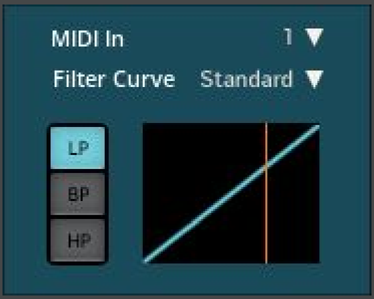
Filter

Chip selection

6581 | 8580

Just like in each SID channel, the button at the top of the Filter panel switches between the 6581 and 8580 chips.

MIDI In



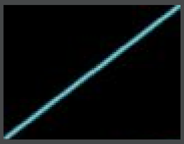
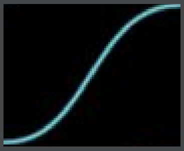
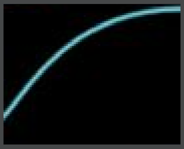
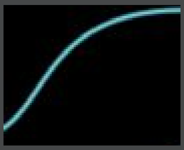
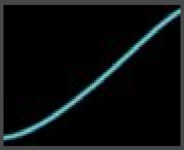
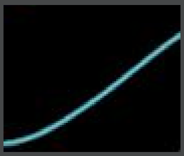
MIDI In 1 ▼
Filter Curve Standard ▼

The MIDI In control sets the MIDI control channel of the Filter. The Filter's envelope and LFO will only be retriggered when a note has been received on its MIDI In channel. Let's say you have a lead sound on SID channel 1, but a bass on SID channel 2 that uses the filter. SID channel 2 has its MIDI In set to 4, which means that the Filter's MIDI In must also be set to 4 so that the filter will retrigger on each bass note.

Filter Curve

The Filter Curve gives some flexibility as to how the filter cutoff operates to emulate the variety of curves found in the 6581 chips. It doesn't apply to the 8580. If the 8580 Chip is selected, the Filter Curve will be set to "8580" and cannot be changed. The 8580's curve is the same as the Standard curve listed below.

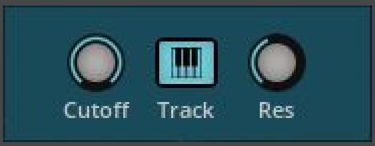
The range of filter curves was determined by the great work done by Antti Lankila in his analysis of the 6581's distortion. The available filter curves, which are based on information taken from his website, are as follows:

Standard	This is the most natural sounding curve and is the same as that found on most synthesizers.	
Avg SID	This is a general curve that is representative of the majority of 6581 chips.	
Follin	This curve is one that works very well for Tim Follin's music.	
Galway	This represents the curve that was present on Martin Galway's Commodore 128.	
Strong	This curve is biased towards the low frequencies, but still maintains a reasonable range.	
Extreme	This one is among the more extreme examples and is heavily biased towards the low frequencies to the detriment of the high frequencies.	

Modulation Controls

The modulation control panel is an instance of the Common Modulation Panel and so operates in exactly the same way as those found in the SID channels. The LFO Depth control goes from 0.0 to 1.0, but that produces an LFO that goes from -1.0 to 1.0, which is larger than the range of the cutoff value. This is to allow further flexibility in control of the cutoff value. For a simple LFO that sweeps the entire cutoff range, set the LFO Depth to 0.5 and the Cutoff value to 0.5.

Cutoff and Resonance



The two knobs at the bottom allow you to adjust the cutoff frequency and resonance. The Track button works in a similar way to the equivalent button in the Step Table. When enabled, the filter cutoff will increase or decrease according to the incoming MIDI note on the Filter's MIDI channel.

External Filter Input

Using the standard routing capabilities of your DAW, you can route a mono audio signal into inSIDious to have it be affected by the filter.

Options

In the bottom-right corner of inSIDious, in the output scope is the Preferences button: ⚙. Clicking this will show the options list. The available options are:

Enable MIDI velocity
Enabling this switch will cause all SID channels to adjust the output volume according to the incoming MIDI note velocity. This option is saved with each individual snapshot that you might store.

Enable scopes
While enabled, the oscilloscopes will all remain in operation. If disabled, the oscilloscopes will switch off saving some CPU. This value is not saved with user snapshots.

Remove circuit noise
The real SID is quite noisy and inSIDious recreates the clicking and noise that occurs under certain circumstances. The click that occurs when new notes are played can even cause a tonal change to the 6581 filter, causing it to act as if a short decay is set on the filter envelope. Although these contribute to the accuracy of the emulation, you might not want to hear them in your music, so you can optionally switch it off here. This option is saved with each individual snapshot.

Global Tuning
By clicking and dragging on the value here, you can raise or lower the root pitch of the whole instrument. This value is not saved with user snapshots.

For Owners Of The Full Version Of Reaktor

By default, sending a Program Change to inSIDious will change the settings of all three channels and the filter at once. However, if you own the full version of Reaktor, you can modify inSIDious so that you can send a Program Change to each individual SID channel. This allows you to change instruments on each channel independently and open the door to being able to, among other things, recreate actual Commodore 64 music.

1. Right-click on the coloured bars at the top of channel 1 and choose "Instrument Properties".
2. In the left panel, go to the "Function" tab.
3. In the SNAPSHOTS section of the panel switch on "Recall by MIDI"..
4. Do the same for channel 2 and 3 and for the filter. For the filter, just right-click to the right of the 6581 / 8580 button to choose "instrument Properties".

You can now store embedded snapshots for the individual channels and filter (see the Reaktor manual) and recall them by sending a MIDI Program Change to their selected MIDI channel.

Appendices

A: inSIDious / Real Hardware Differences

There are some elements of the SID that are not very conducive to making music. Rather than emulate these issues, forcing the user to find ways around them, inSIDious deliberately does ***not*** emulate the following:

6581

Oscillator Leakage: During the release phase of the volume envelope, the volume sometimes never reaches zero, allowing the last sound to be heard at low volume continuously until the channel is properly reset.

Filter Curves: The nature of the filter curve problem means that there are potentially an infinite variety across all chips, although there is a great deal of commonality.

Combined Waveforms: The same problem that affected the filter curve also affected the combined waveforms to a much lesser degree. On most 6581 chips, you would be very hard-pressed to hear any differences between chips, but in some of the outlier chips the difference is noticeable. In some extreme cases, a waveform might even sound an octave higher than usual.

Background Noise: The real 6581 produces a lot of background noise, a large proportion of which is due to the ability to route external audio through the filter. inSIDious allows you to route audio through its filter, but this feature obviously does not introduce any noise.

8580

Linear Filter Curve: Because the filter curve on the 8580 is linear, small changes in the cutoff value at low frequencies have a much greater effect than those at high frequencies. The 8580 filter curve in inSIDious instead uses a more natural sounding logarithmic curve (identical to the "Standard" curve when inSIDious is in 6581 mode).

Both

Per-channel Volume: The real SID only has a global volume and does not allow the volume of each channel to be set independently. Also, inSIDious has an option to respond to MIDI velocity for dynamic control of the channel's volume.

Filter Curve Offsets: At certain cutoff values, there is a jump back in the cutoff frequency of the filter.

ADSR Bug: Under certain circumstances, the real SID might not retrigger the volume envelope correctly.

Noise Generator Lockup: When using the noise waveform combined with any other, the noise generator can lock up, causing it to produce only silence until the channel is reset.

TEST Bit: When the TEST bit is set, it will reset the oscillator to zero and hold it at zero until it is cleared, at which point the oscillator will be restarted. inSIDious uses the TEST bit feature in the Step Table as if the bit was set and then cleared in quick succession, causing a reset of the phase of the oscillator waveform.

B: Control Change Assignments

5	Portamento Time
7	Volume
20	6581/8580
21	Filter On/Off
22	Table Speed
23	PWM Centre
24	Octave
25	Note
26	Fine
27	Volume A
28	Volume D
29	Volume S
30	Volume R
70	6581/8580
71	Resonance
74	Cutoff
75	LP On/Off
76	BP On/Off
77	HP On/Off

C: Volume Envelope Values

Attack Time	**Decay Time**	**Release Time**
0 = 2 ms	**0 = 6 ms**	**0 = 6 ms**
1 = 8 ms	**1 = 24 ms**	**1 = 24 ms**
2 = 16 ms	**2 = 48 ms**	**2 = 48 ms**
3 = 24 ms	**3 = 72 ms**	**3 = 72 ms**
4 = 38 ms	**4 = 114 ms**	**4 = 114 ms**
5 = 56 ms	**5 = 168 ms**	**5 = 168 ms**
6 = 68 ms	**6 = 204 ms**	**6 = 204 ms**
7 = 80 ms	**7 = 240 ms**	**7 = 240 ms**
8 = 100 ms	**8 = 300 ms**	**8 = 300 ms**
9 = 250 ms	**9 = 750 ms**	**9 = 750 ms**
10 = 500 ms	**10 = 1.5 s**	**10 = 1.5 s**
11 = 800 ms	**11 = 2.4 s**	**11 = 2.4 s**
12 = 1 s	**12 = 3.0 s**	**12 = 3.0 s**
13 = 3 s	**13 = 9.0 s**	**13 = 9.0 s**
14 = 5 s	**14 = 15 s**	**14 = 15 s**
15 = 8 s	**15 = 24 s**	**15 = 24 s**

D: Further Reading

General Synthesis

https://en.wikipedia.org/wiki/Pulse-width_modulation

https://en.wikipedia.org/wiki/Ring_modulation#Integrated_circuit_methods_of_ring_modulation

https://en.wikipedia.org/wiki/Subtractive_synthesis

SID Technical information

https://en.wikipedia.org/wiki/Robert_Yannes

http://sid.kubarth.com/articles/interview_bob_yannes.html

https://bel.fi/alankila/c64-sw/index-cpp.html

http://sid.kubarth.com

https://sourceforge.net/projects/sidplay-residfp/

SID Music

https://c64audio.com

https://hvsc.c64.org

https://remix64.com

https://www.adamdawes.com/retrogaming/rg_04_pias.html

Other Related

https://www.goodreads.com/book/show/39751835-bits-and-pieces

https://www.goodreads.com/en/book/show/412006.On_the_Edge

E: About the contributors

Chris Huelsbeck

Chris Huelsbeck is an award winning composer and sound designer best known for his stunning work in the video games industry from the late 80s onwards. He started his career at a young age by programming his own custom sound system for the Commodore 64. Since then, Chris has continued to push the envelope with every new generation of sound hardware.

You can find Chris's music at http://chrishuelsbeck.bandcamp.com/ and can support him directly at

https://www.patreon.com/chris_huelsbeck

Fabian Del Priore

Fabian Del Priore, aka Rapture, is a composer, producer, and remixer best known for his game music compositions such as Giana Sisters: Twisted Dreams, Giana Sisters DS, Bubsy The Woolies Strike Back, Extreme Assault, Cultures 2, to name a few. He is a long-time collaborator with Chris Huelsbeck with whom he has worked on many game soundtracks.

His tracks also appear on popular video game music/remix CDs and demoscene compilations, such as Immortal (Amiga Remix CDs), RevivalST (Atari ST Remix CD), Remix64 (C64 Remix CD), The Sound of Scenesat among many others.

Check his Patreon out here to support him: https://www.patreon.com/fabian_del_priore

Jason Page

Jason Page has been working in the games industry since 1988. He began his career as a programmer and musician on the Commodore 64, but continuously developed his skillset to the point where he became a defining part of Sony's Playstation audio division. While at Sony, not only did he produce music for a multitude of high-profile games, but also created the Multistream audio system that was used in thousands of PlayStation titles. In 2015, he decided to go back to his roots and write SID chip music once again, just to see what he could coax out of the chip using his extra knowledge and understanding. His recent music productions are highly acclaimed in the C64 demo scene and his first ever C64 demo, entitled "Threshold" even placed second in the "Mixed Demo" category at Flashback 2019.

You can hear some of Jason's productions at https://soundcloud.com/noothermedicine

LMan

LMan, aka Markus Klein, started making electronic music in 1990 on his C64. Although he progressed onto the Amiga and then PC DAWs, he never strayed far from the SID and is the co-founder and operator of remix64.com; a web community dedicated to remixes of classic game music. In 2015, LMan returned to the SID chip and actively entered the demoscene, where he quickly stood out as a true SID virtuoso with his incredible inventive sound design and sample usage. In 2020, he sailed past his Kickstarter goal to release a full SID album on vinyl and cartridge. Markus is a member of distinguished scene groups like MultiStyle Labs, Maniacs of Noise, Censor Design, Performers and TRSi, and is also an accomplished and internationally recognised pencil artist.

You can find Markus's collected art and musical works at https://markus-klein-artwork.de

Jammer

Jammer, aka Kamil Wolnikowski, has been a driving force in the C64 demo scene since 2002. He is a hugely prolific and celebrated composer, with each of his releases demonstrating new techniques that coax previously unheard sounds from the SID chip.

You can see his complete demoscene works at https://csdb.dk/scener/?id=8105 and listen to some of his full music productions on his soundcloud page at https://soundcloud.com/kamil-wolnikowski

TDK

TDK. aka Madfiddler, aka Mark Knight, is a veteran chip and game musician (and accomplished violin player) who began his career creating music for demo groups on the Amiga. From there he found his way into creating music for games and has worked on a multitude of high-profile game titles for companies such as Mindscape, Bullfrog, EA, and Codemasters. He has always been passionate about chip music, especially where the Commodore 64 and SID chip are concerned and continues to produce extraordinary chiptune albums and perform live chiptune sets across the globe.

Check out Mark's music at https://www.flitkillsmoths.co.uk or https://marktdkknight.bandcamp.com or find more information about his audio design specialties at https://www.sonicfuel.co.uk.

Martin Galway

Martin Galway is one of most renowned and revered SID composers of all time. He is a true pioneer of video game music and a developer who programmed the SID every day between 1984 and 1988 as part of his job creating Commodore 64 games. He is one of the earliest developers who were able to forge a profession by making video game music, creating what we now call "the 8-bit sound". Martin was first to create music with fast arpeggiation, which has become one of the most distinctive and defining techniques of the chiptune genre. He was also the first to use sample playback, non-standard waveforms, and filter envelopes in SID music. These pioneering techniques were used to create incredible, sometimes even haunting, emotive scores that always left a lasting impression on the listener.

Rob Hubbard

Anyone even remotely familiar with the SID has likely heard of Rob Hubbard. He is another of the most renowned and revered SID composers, having created music for a whopping 67 C64 games between 1985 and 1988. His coveted self-coded music player allowed the usage of various clever techniques that were very difficult to reproduce by others and Rob's music was often easily recognisable due to its sound. His technical prowess coupled with his enviable music ability resulted in many people buying games solely because his music was on them. Rob received an honorary Doctorate in Music from the University of Abertay in 2016. In 2019 and 2020 he was closely involved in the 8-bit Symphony project, where he re-arranged some of his classic music for live orchestra performance. You can hear the amazing results at https://www.8-bit-symphony.com or find out everything you could ever want to know about Rob from Project Hubbard at https://c64audio.com/collections/project-hubbard

F: Thanks to

Martin Galway

Rob Hubbard

Chris Huelsbeck

Jason Page

Markus Klein

Kamil Wolnikowski

Fabian Del Priore

Mark Knight

Fabio Marinelli

Chris Abbott

Jeroen Tel

Sascha Zeidler

Antti Lankila

Andrew Aversa

Dickie Chapin

Mario Krušelj

Liliana Pantoja

Native Instruments

...and Tycho…