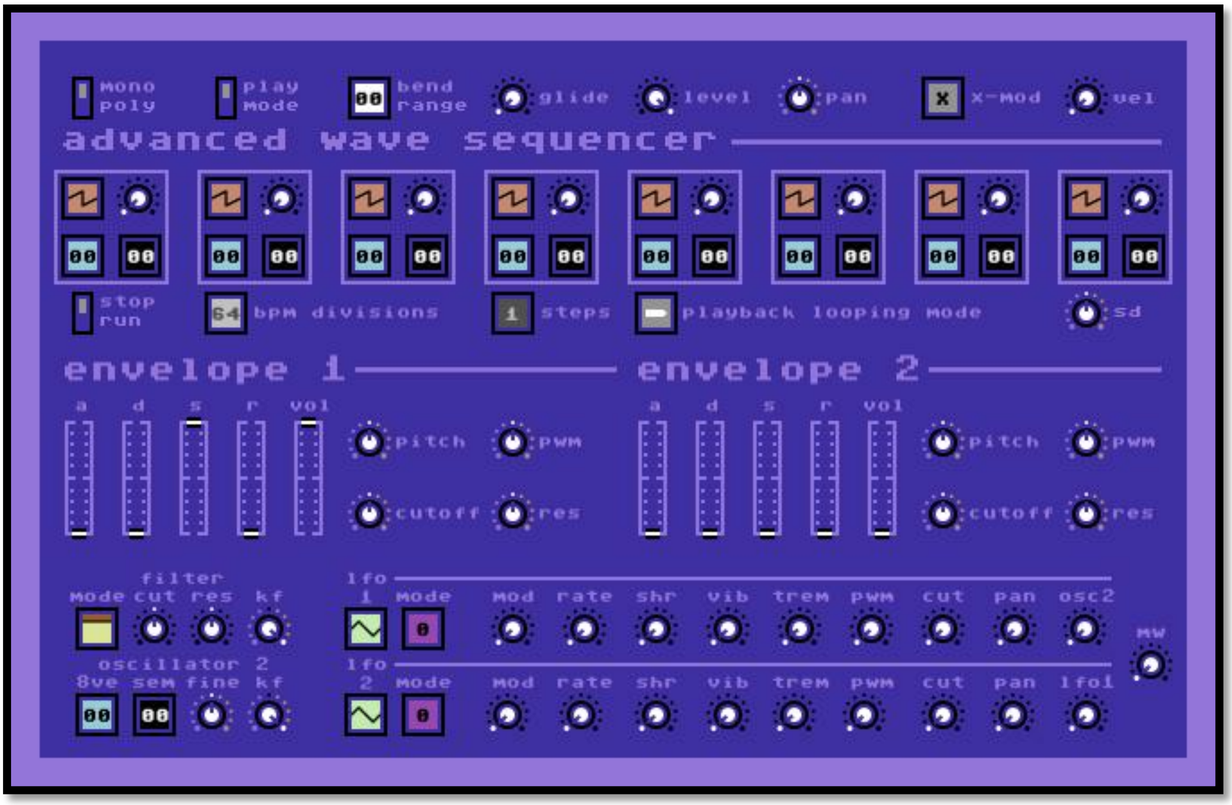
simple-media.co.uk

retro-gsx
user guide & reference manual



Contents

‘retro-gsx user guide and reference manual’

By: Saul Cross (copyright Saul Cross - simple-media 2011)

Introduction

Welcome to this user guide and reference manual for retro-gsx. Over the next few pages you will read about the background of the retro-gsx vst and discover how to use its many features to create an authentic and impressive array of retro sound chip style sounds.

Inspiration

The inspiration for retro-gsx came from a love of computer game soundtracks from the early 1980s through to the early 1990s. Unlike modern game soundtracks that leverage CD and DVD quality audio in a live mixing environment to create music and sound effects, early games used specialist hardware built into the computer known as sound chips to create all their audio. These sound chips had significant limitations such as a low number of voices limiting polyphony and a limited number of source sounds and sound shaping features which in turn would limit the types of sounds that could be created. This meant that musicians (who were often as much, if not more, a programmer than a musician) relied upon their programming ingenuity to create their musical output.

Popular sound chips included the Atari POKEY found in the Atari 400 and 800, General Instuments AY-3-8910 and AY-3-8912 found in the Amstrad CPC 464, a number of MSX computers and later models of the Sinclair ZX Spectrum, and the legendary MOS SID 6581, 6582 and 8580 chips found in the Commodore 64 and 128 family of home computers. Due to the popularity of the Commodore 64 (the single highest selling home computer to date) the SID chip is one of the most widespread pieces of sound hardware ever created.

POKEY

Pokey is an abbreviation of Pot Keyboard Integrated Circuit and was a digital I/O chip found in Atari's 8-bit home computers and also used in a number of their arcade machine boards. It provided 4 digital audio channels that could be used independently (8-bits per channel) or could be combined, as either 2 16-bit channels or 1 16-bit and 2 8-bit channels. Each channel could generate either a square wave with variable duty cycle (a pulsed wave) or band limited noise (using a pseudo-random number generator). In addition each channel had independent control over volume (4-bits allowing for 16 volume levels). There were no envelopes or filters so all shaping of the sound generated had to be performed in software by feeding different values into the volume, frequency, duty cycle, and noise registers of the channel while it was sounding. Tuning was generally not good.

AY-3-8910 and AY-3-8912

The AY family of sound chips or Programmable Sound Generators (PSG) as they were known appeared in a wide range of home computers, arcade machine boards and home consoles as a result of which there were probably more AY chips than SID chips produced. The chip produced three audio channels which could generate either square waves or pseudo-random noise, the square wave and noise could also be mixed and the sound could also be modulated at audible frequencies to create additional harmonics. In addition to the mixing and modulation available the sound could be further shaped using an ADSR envelope generator which had the interesting ability to be able to rapidly repeat or to invert part of the envelope cycle, again at audible rates creating extremely hard attack or a buzzer type sound. Tuning was better than POKEY but still limited to only a 12-bit frequency table so some notes could be sharp or flat as a result.

Sound Interface Device (SID)

The SID chip was designed by Bob Yannes as part of the custom hardware developed for the commodore 64 (Bob Yannes was also involved in the development of the VIC2 chip which provided the Commodore 64's graphics). Yannes decided early on in the design of the chip that he wanted to produce a professional quality sound device that could accurately reproduce frequencies and had many features associated with synthesizers of the time. The original intention was to create a 32 voice chip using a divide down oscillator but this was not possible in the time and budget available so the design was scaled down to produce a chip with 3 independent phase accumulating oscillators (digital wavetable oscillators although the wavetable had to be generated on the fly), 3 independent DCAs (ADSR envelope generators and level for each voice), a single analogue filter to which each voice could be independently routed and additional features including ring modulation and oscillator synchronization.

Each oscillator is capable of producing a sawtooth wave, a triangle wave, a variable duty cycle square wave (pulse wave) and band limited pseudo-random noise. More than 1 wave could be played simultaneously with the outputs being logically ANDed together, although the results from this were variable and combining noise with any other waveform could lock up the accumulator until a hard reset was performed. Tuning used a 16-bit frequency look up table which was considerably more accurate than other chips at the time. The frequency range of the chip covered almost 8 octaves (from around 16 to 4000 Hz).

The ADSR envelope used 2 bytes allowing 4-bits for each of attack, decay, sustain and release. The values again used a look up table for timings allowing for the duration of attack, decay and release to differ in order to create greater flexibility. The sustain level could be one of 16 values as could the overall volume level allowing for a wider variation in output level overall. All values could be modulated in cycle allowing for a much greater amount of control.

The filter was a genuine analogue component, a 1-pole state variable filter with a 12 dB per octave roll-off which had bypass, low-pass, high-pass, band-pass (6 dB per octave) and notch (band-reject again 6dB per octave) outputs. The outputs could be combined in the same way as the waveforms could.

Any voice could be ring modulated by the next voice’s output; the signals were multiplied then added to create the new signal (the oscillator being modulated became forced to a triangle output). Each oscillator could also be synchronised with the next to create more complex harmonic content. Pulse width for the pulse wave could be continuously varied and it is this pulse wave modulation that created the most characteristic sound associated with the SID chip.

In addition the output of oscillator 3 could be muted and read internally providing a source for LFO and audible frequency modulation effects.

In all the specification of SID was like no other sound chip of the time, rivalling many professional synthesizers at a fraction of the cost and under computer control could produce an impressive array of instrumental and non instrumental sounds. So effective and powerful was the SID chip that many consider that it is the SID chip and the quality of sound and music that talented musician programmers could produce with it rather than any other factor that was responsible for the sale of many Commodore 64 home computers.

Masters of the Chips

It was in the hands of talented programmers and musicians that these chips really shone and it is because of their work that the ‘chip sound’ is still popular today. It is as much the work of these composers as it is the chips themselves that insired the creation of retro-gsx. Artists such as Rob Hubbard, Martin Galway, David Whittaker, Tim Follin, Fred Gray, David Dunn, Jeroen Tel, Chris Huelsbeck, Martin Walker and Ben Daglish created a whole new musical culture by exploiting the limited power of such chips and pushing the boundaries of musical and audible creativity.

The retro-gsx VST Instrument

This new retro-gs instrument has been created over a long period, not to replace but to compliment the older retro-gs01 and retro-gs02 VST instruments. While the older instruments work in a way similar to most other virtual analogue instruments and are programmed in a traditional way, the aim was to create an instrument that was far closer not only in sound but also in the way it was programmed to a sound chip based instrument.

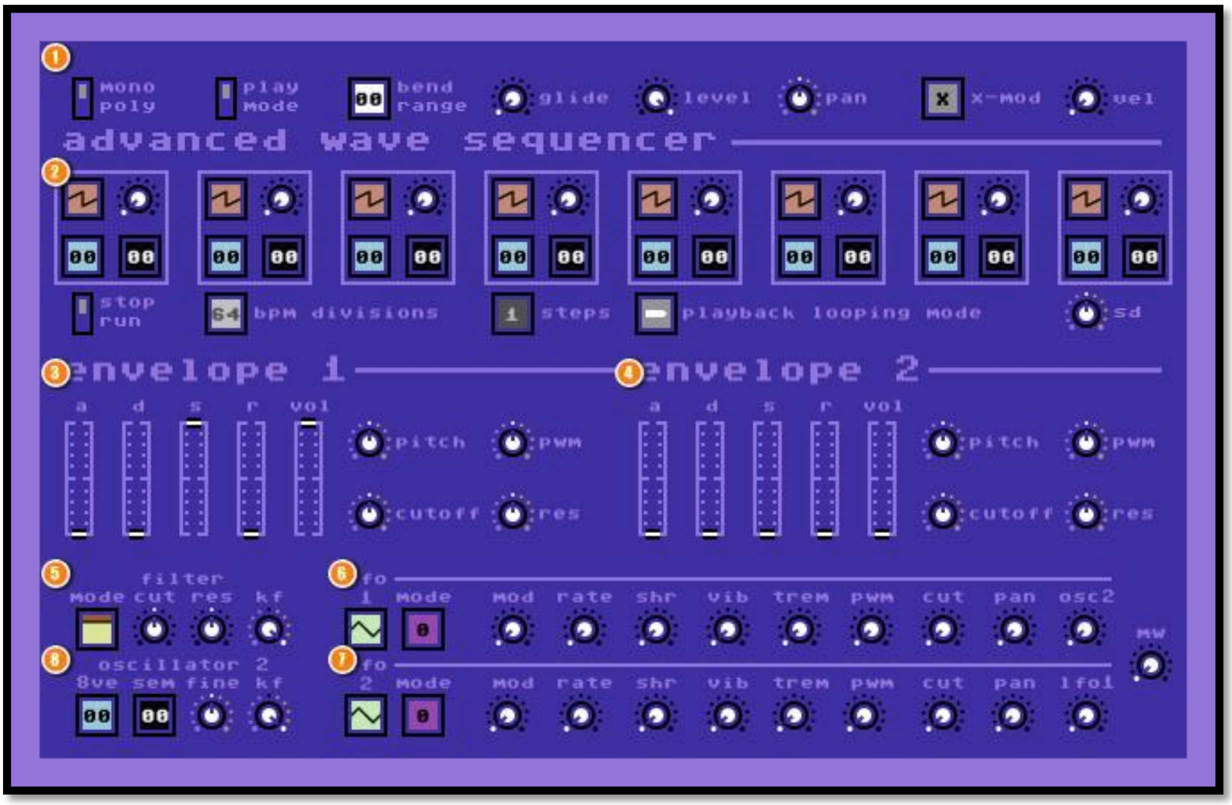
The end result is a single voice instrument with just one audible oscillator which recreates most of what is possible using a single SID voice (retro-gs01 is a more typical two oscillator virtual analogue instrument and retro-gs02 uses the same engine to create a set of drum/percussion sounds).

In order to get more out of the limited architecture of sound chips, programmers would often rapidly change the frequency of a voice, its waveform or some other parameters in order to create new complex sounds, or to create richer harmonies or chords. Early play routines such as Rob Hubbard’s music routine would use a set of specifically coded effects such as a drum that would play a short burst of noise at the beginning of a sound then switch to another waveform or arpeggios that would rapidly switch between two notes. This meant that each music routine would be slightly different and the approach was similar to creating patches in a traditional instrument. Later music routines incorporated a system for playing back rapid sequences of frequency, waveform and other parameter changes using a fixed routine that read data from a so called wave table. A wave sequencer has been built in to retro-gsx that allows the user to program a series of frequency changes, waveform changes and changes to pulse width, which can recreate the effects possible using wave table based play routines.

The engine in retro-gsx was built from scratch and also uses a couple of additional techniques to create a sound that is closer to that of the SID chip. It is not an accurate emulation of the SID chip although it does copy the routing, but does closely resemble the sound of SID.

Interface Overview

The interface uses a combination of icons, control knobs, switches and sliders to access the full range of parameters available. These are broken down into a number of sections as follows:



1 global control to set mono or poly mode, retrigger mode, pitch bend range, glide (portamento) rate, overall level, initial panning, cross modulation mode (off, ring modulation, synchronization and both) and velocity sensitivity;

2 advanced wave sequencer to set waveforms (sawtooth, triangle, pulse or noise), tuning (octave and semitone) and initial pulse width for each stage in the wave sequence, plus whether the sequencer is running, running speed, the length of sequence, playback options and ‘skydive’ function;

3 envelope 1 – primarily the DCA ADSR envelope but can also control pitch, pulse width, filter cut-off frequency and resonance;

4 envelope 2 – does not affect voice volume but provides additional control over pitch, pulse width, filter cut-off frequency and resonance;

5 filter control sets the filter mode (bypass, low-pass, high-pass, band-pass or notch), initial cut-off frequency, resonance and key follow (can also be inverted);

6 LFO 1 (low frequency oscillator) has controls for waveform (sawtooth, triangle, square or noise), mode (off, on and ‘sample and hold’), overall modulation level, frequency (rate), ‘sample and hold’ rate, vibrato (frequency modulation), tremolo (amplitude modulation), pulse width modulation, filter cut-off frequency, panning and frequency modulation of the inaudible second oscillator;

7 LFO 2 has controls for waveform (sawtooth, triangle, square or noise), mode (off, on and ‘sample and hold’), overall modulation level, frequency (rate), ‘sample and hold’ rate, vibrato (frequency modulation), tremolo (amplitude modulation), pulse width modulation, filter cut-off frequency, panning and frequency modulation of LFO 1;

8 Oscillator 2 is an inaudible oscillator fixed as a triangle waveform that is used for cross modulation with the audible oscillator to create ring modulation and synchronization effects; it has controls for tuning (octave, semitone and fine tune) and key follow (can also be inverted).

Basic Sound Creation

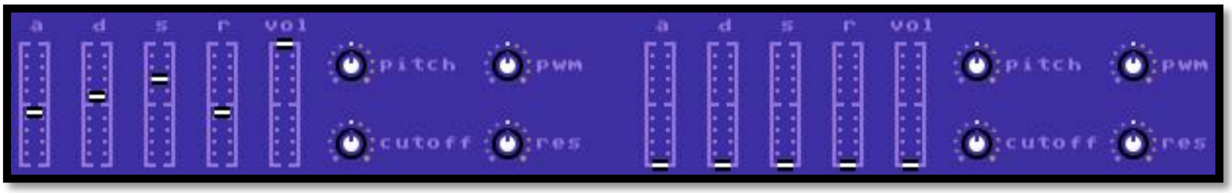
Sound creation begins by selecting a waveform. In this case if the advanced wave sequencer is set to stop rather than run then the first step in the sequence provides the setting for the wave form and the initial settings for tuning and pulse width (if applicable). If the advanced wave sequencer is set to run, then the waveform, tuning and pulse width will vary as the sequence runs. The available waveforms are sawtooth, triangle, pulse wave and band limited noise (the noise is affected by the frequency it is played at as if it was being passed through a low resonance band-pass filter).

The waveforms are represented by a series of icons, clicking on the current waveform icon will cycle through to the next waveform. The waveform icons are colour coded making it easy to see which waveform is selected at glance (red = sawtooth, green = triangle, blue = pulse and gray = noise).

advanced wave sequencer icons

The cyan numbered icon is the current octave setting (range is -2 to +2 octaves) and the black numbered icon the current semitone setting (range is -6 to +6 semitones). Finally each step includes a small rotary knob which is used to set the initial pulse width; anything from a square wave to a very narrow pulse.

The sound is then shaped using envelope 1 to control how the sounds volume changes over time. This is a standard ADSR envelope with settings to alter the rate of attack, decay and release and a single sustain level plus an overall volume control. In addition to this the output of envelope 1 can be used to control voice frequency, pulse width, filter cut-off and resonance. The controls for this are set to 0 when in the centre and can therefore be set to have a negative or positive effect on the initial values. Envelope 2 offers the same controls but does not affect volume.



envelopes 1 and 2

The sound can also be modulated using the 2 LFOs each of which can be used to modulate frequency, volume, pulse width, filter cut-off frequency, panning and also the inaudible second oscillator and LFO 2 can modulate LFO 1. The ‘mw’ control situated to the right of the LFO controls is used to determine how much vibrato the mod wheel or after-touch will add to a sound allowing for more dynamic playing (this works by sending the LFOs at a fraction of their ‘mod’ level; multiplied by either the mod wheel or after-touch control value and the mw setting so it is independent of any other vibrato).

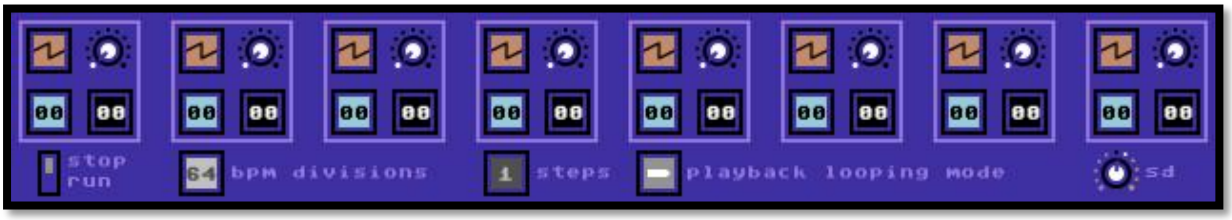
LFOs 1 and 2

The sound now passes through the filter section which allows control over the type of filter (bypass, low-pass, high-pass, band-pass or notch), the filter’s initial cut-off and resonance and filter cut-off key tracking (the control for this is set to 0 when in the centre and can therefore be set to have a negative or positive effect). The filter modes are selected by clicking on the yellow and brown filter icon which will cycle through to the next mode.

filter controls

Wave Sequencing

At the heart of retro-gsx is the advanced wave sequencer; it is this feature that gives retro-gsx real control over an authentic chip sound.



advanced wave sequencer

As can be seen the advanced wave sequencer offers up to 8 steps, each of which controls the waveform, initial tuning (octave and semitone) and pulse width of the audible oscillator. Complex instrument sounds, arpeggios (chords) or rhythmic sequences can be built up by carefully setting these parameters.

The speed of playback can be controlled in relation to the host (DAW) tempo, using the bmp division icon the sequence can step through 64, 32, 24, 16, 12, 8, 6, 4, 3, 2 or 1 time for each beat in the host tempo.

The number of steps can be set from 1 up to 8.

The playback looping mode has 4 settings, (single run – where the last settings in the sequence will be held indefinitely, single run and stop – where the note will no longer sound after reaching the end of the sequence, loop – which loops forward continuously and ping-pong – which loops back and forth).

Finally there is the 'sd' or 'skydive' control. This was added for a specific purpose to recreate an effect where the note rapidly switches between a fixed pitch and a falling pitch (think 'Monty on the Run'). This has been referred to as Rob Hubbard's 'skydive' effect as it sounds like somebody screaming as the fall away from a plane. This works by sending the output of envelope 2 and the mod level output of the LFOs to only every second step in a sequence's frequency (steps 2,4,6 and 8 are affected). It can be used to create a wide range of effects that would otherwise be impossible.

Cross Modulation

In order to offer the same cross modulation options (ring modulation, synchronization and a combination of the two) as a real SID chip, retro-gsx features a second, inaudible oscillator that is preset to a triangle wave and can be independently tuned and modulated in relation to the audible oscillator.

oscillator 2 controls

It offers controls for tuning (octave, semitone and fine tuning) and a control for key tracking which can be inverted (it is set at 0 when it is in the centre therefore it can have a negative or positive effect). It is important to note that in synchronization modes it will be the frequency of the second oscillator that is heard and the main oscillator will provide the changes to harmonic content.

Using ring modulation it is possible to create a wide range of harsh metallic or screaming tones as well as much more subtle glassy, bell like sounds.

Synchronization creates coarse harmonic sweeps in sounds when the main oscillator has its frequency modulated relative to oscillator 2. This is very good for creating hard synthetic sounds or gritty electric guitar like sounds.

Presets

There are 64 presets included with retro-gsx, the table below shows you a complete list of presets in each of the preset categories included:

Category	*Presets*
PER - percussion	1.Hard Kick, 2.Hi-Hat, 3.Snare, 22.Trad Kick, 23.Trad Kick Plus, 50.Block 1, 51.Block 2, 52.Shaker, 53.Just Noise
BAS - bass	4.Filtered Saw, 20.NES Style 1, 24.Most Basic, 25.Pulse Bass, 29.Zipper, 32.Big ‘n’ Fat, 41.Soft ‘n’ Low, 60. Bouncer, 61. Fat Bouncer
LED - Lead	5.Smooth Pulse, 6.Pulse Chiff, 15.Synchronizations, 17.Saw Fiddle, 18.Simple Flute, 19.Simple Reed, 21.NES Style 2, 26.Blue Peter, 28.Twang, 33.Quitar, 44.Buzzing, 47.Pulse Reed 1, 48.Pulse Reed 2, 49.Pulse Reed 3, 59.Distraughted
SYN - synthesizer	7.Rattle, 42.Bubble, 43.Ice Crystals, 57.Noise Drops
ARP - arpeggio	8.Major Chord, 9.Minor Chord, 54.Slow Build
SFX - sound effects	10.Shimmer, 14.Ring Modulated Saw, 16.Sky Diver, 34. BBC One, 37. By the Sea, 38.Wind Rush, 39.Heavy Rain, 40.Thunder Stuff, 45.Birds, 58.A Haunting
PAD - pad sounds	11.One Voice Strings, 27.Shades of Gray, 35.Sassy Brass, 36.Classy Brass, 62.Super Smooth
SEQ - sequences	12.Gate Like Sequence, 55.Electronica
KEY - keyboard	13.Tinker Bell, 30.Perhapsicord, 31.Eee Pee, 46.Fairground, 56.Eastern Promise, 63.Clarity
other	64.Initialized Patch

These presets are by no means indicative of the limits of retro-gsx and are included just to give an idea of some of the sounds that are possible and how certain effects might be achieved.

Limitations and Known Problems

As retro-gsx is not a perfect copy of a SID chip it does have some limitations and known problems that users should be aware of, although there are workarounds to these as follows:

1. waveforms cannot be combined – unlike the SID chip retro-gsx will only allow for one waveform to be played at any one time, in many cases a similar kind of effect can be achieved audibly by careful use of cross modulation and the LFOs;

2. filter types cannot be combined – unlike the SID chip retro-gsx will only allow for one type of filter output at any one time, in most cases this can be fixed by post filtering using a host applications EQ settings;

3. the sound is not as distorted as a real SID chip – of course this is not a real SID chip but distortion can be added as a post effect in most host applications and while it is easy to do this it is much harder to remove distortion from a real SID chip (in most cases setting the level high enough for a small amount of clipping recreates the SID chip distortion quite nicely);

4. it sounds different to the SID chip in one of my Commodore 64s – then again the SID chip in the other Commodore 64 also sounds different; it will not produce a perfect copy of any given SID chip but no 2 SID chips sound the same anyway – retro-gsx has a bit of character of its own;

5. notes stick sometimes when using an external MIDI arpeggio – it is advisable to use the advanced wave sequencer for arpeggios and pseudo-chords with several instances used for different chord types, e.g. one for major, one for minor, etc.

If you experience any problems with retro-gsx please contact me via the following e-mail address and I will do my best to find a solution:

saulc@saulc.karoo.co.uk