VIRTUAL ANALOG OSCILLATOR HARD SYNCHRONISATION: FOURIER SERIES AND AN EFFICIENT IMPLEMENTATION

Joseph Timoney, Victor Lazzarini, and Matthieu Hodgkinson
Sound and Music Technology research group,
NUI Maynooth
Maynooth, Ireland
joseph.timoney@nuim.ie
victor.lazzarini@nuim.ie

Jari Kleimola, Jussi Pekonen, and Vesa Välimäki,

Dept. of Signal Processing and Acoustics,
Aalto University
Espoo, Finland
jari.kleimola@aalto.fi
jussi@pekonen.cc
vesa.valimaki@aalto.fi

ABSTRACT

This paper investigates a number of digital methods to produce the Analog subtractive synthesis effect of 'Hard Synchronisation.' While the original effect is produced by an explicit waveform phase reset, other approaches are given that produce an equivalent output. In particular, based on measurements taken from a real-analog synthesizer, a comb filtering model is proposed. This description ties in with earlier work but here an explicit structure is provided. This filter-based approach is then shown to be far more computationally efficient than the synchronisation by phase reset. This efficiency is at a minor cost as it is shown that it has a minimal impact on the sonic accuracy.

1. INTRODUCTION

Oscillator synchronization (Hard Sync) is an important user option that any digital subtractive synthesizer should offer. Synchronization was originally developed for analog synthesizers to counteract the frequency drifting that can occur between voltage controlled analog oscillators, as too much drift can make the musician appear to be out of tune. Oscillator synchronization comes in two forms: Hard sync and Soft sync. Of the two, Hard Sync is the more striking effect. It is noted for its dynamic, expressive, screaming quality that is excellent for creating remarkable lead and bass sounds ([1], [2], and [3]). Hard synchronization is normally associated with sawtooth oscillators and requires two oscillators to work. These are termed as the Master and Slave respectively. For the effect to be noticeable, the Slave oscillator should be at a higher frequency than the Master. In essence, hard sync locks the waveshape of the slave oscillator to that of the master, resetting to its initial value 'in sync' with that of the Master as it commences a new period. This ensures that both oscillators are at the same period, eliminating the frequency drift between them.

While algorithms for the bandlimited implementation of Hard Sync exist ([4, [5], and [6]), there is no ideal version against which to benchmark it. Thus, it would be very useful to have an Fourier series/additive synthesis description of the Hard Sync waveform. Having such a description it should be able to suggest a way by which a digital implementation could be made using tailored digital elements. This should lead to a less complex, and therefore cheaper, way to implement it for virtual reproductions of analog synthesis operations. A lower implementation cost would confer benefits such as greater polyphony from the virtual synthesizer. In contrast, although attempting to directly simulate a particular analog circuit design offers more modeling accuracy, it normally results in an algorithm that is computationally intensive because of nonlinear circuit elements, requiring a significant oversampling factor to operate correctly, see [7], [8], [9], and [10] for example.

In this paper, we will therefore present an additive synthesis /Fourier series description of Hard Sync followed by an efficient implementation using standard DSP elements. The paper is organized as follows: Section 2 examines the Hard Sync effect using examples that were measured from actual analog synthesizers. Derivations for the Fourier series will also be given in Section 2. This will be followed by a delay-line filter-based implementation in Section 3. An evaluation of its computational efficiency relative to a recent alternative implementation reset-based implementation will be carried out in Section 4. Additionally, the accuracy of the delay-line approach will be evaluated with respect to the Fourier series description and the alternative implementation. One benefit of the filtering approach is that it will not introduce new aliasing distortion components into the signal. Section 5 then completes the paper with a conclusion and some areas for future work.

2. THE HARD SYNC EFFECT

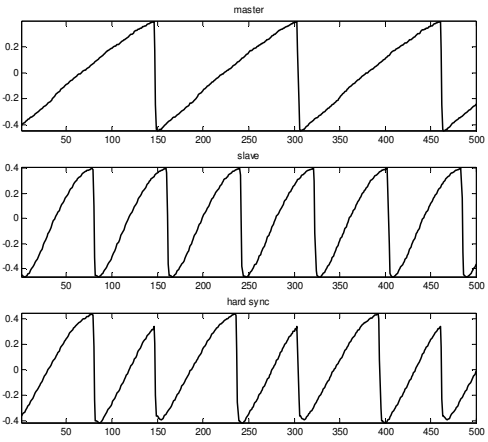

Figure 1: *Example of Minimoog Hard Sync waveform. The top panel shows the Master waveform, the middle panel the Slave and the bottom panel the Hard Sync output.*

Fig. 1 plots an example of oscillator sync from a Minimoog Voyager [2]. The top panel shows the Master, while the middle panel shows the Slave, and the lower panel is the hard sync output. The reset in sync with the Master can be seen in the lower panel. What makes hard sync remarkable, however, is that when the frequency control of the Slave oscillator is adjusted, either manually or using an LFO or envelope, the timbre of the Hard Sync output exhibits a harsh, dissonant quality.

Although the spectrum of the Hard Sync output has harmonics at the same pitch as the Master, the timbral modification is manifested as regularly-spaced formant-like resonances in the spectrum of the Slave. The sound is most sonorous when the relationship between the pitch frequency and the frequency location of the resonances is non-integer. To illustrate, a hard sync output was recorded from the Minimoog Voyager at a sample rate of 44100Hz where the frequency of the Slave is driven by a rising envelope. The spectrogram in Fig. 2 shows the time-frequency profile of the oscillator output and superimposed is a solid white line that shows the adjustment of the frequency control of the Slave oscillator. This spectrogram was computed using an 800-point Chebyshev window, overlapped by 50%.

In Fig. 2 the underlying harmonics of the sync waveform are shown as horizontal grey lines, and the resonances are visible as dark black lines. The actual harmonic frequencies are static through-out, but the resonances move in an almost harmonic relationship following the Slave oscillator's frequency control in a quantized fashion.

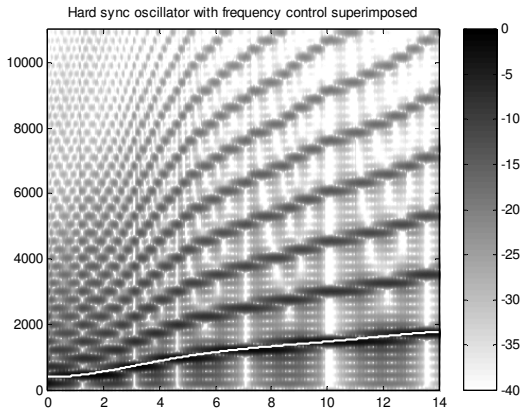

Figure 2: *Spectrogram of hard sync output signal with frequency control adjustment superimposed as a solid white line.*

2.1. Direct Digital Reproduction

A direct digital reproduction of hard sync can be achieved by simply resetting the Slave waveform at the appropriate points in time. This can be illustrated using the following. If the Master waveform is generated using the phase accumulator

$$\theta_{master}(t) = \left(\theta_{master}(t-1) + 2\pi f_{master}\right)\%\left(2\pi\right) \quad (1)$$

where the frequency of the master waveform is given by f_{master} and the modulo operation is denoted by %.

Similarly the phase accumulator for the Slave waveform is given by

$$\theta_{slave}(t) = \left(\theta_{slave}(t-1) + 2\pi f_{slave}\right)\%\left(2\pi\right) \quad (2)$$

The phase accumulator for the hard sync waveform then is generated according to the following conditions

$$\theta_{sync}(t) = \theta_{slave}(t) \quad (3)$$

unless when $\theta_{master}(t) < 2\pi f_{master}$ and the phase is reset according to

$$\theta_{sync}(t) = \left(\frac{f_{slave}}{f_{master}}\right)\theta_{master}(t) \quad (4)$$

A sawtooth can be generated from the accumulated phase by

$$S_{saw}(t) = \theta(t)/2\pi \quad (5)$$

Fig. 3 gives an example of a Master and Slave waveforms and the Sync output using this algorithm. The frequency of the Master is 441Hz, that of the Slave is 723Hz,

for the Hard sync waveform is 441Hz. The effect of the reset is clear on the Hard sync wave and it is clearly in synchronization with the Master.

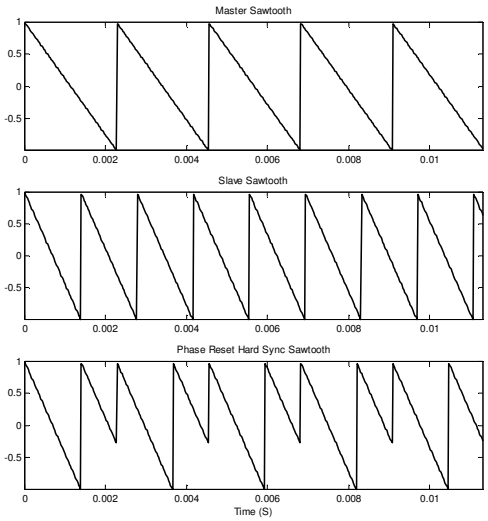

Figure. 3: *Example of Hard Sync. The top panel shows the Master waveform, the middle panel the Slave waveform and the lower panel a Hard Sync waveform synthesized using eqns. (3),(4) and (5).*

However, using this approach the sync waveform is susceptible to the problem of aliasing distortion due to sharp transitions in the waveform. A limiting issue to overcoming this, as raised by [11], was the lack of an expression for the Fourier series of Hard Sync. It is possible to use the phase reset of eqns. (3) and (4) to drive a bank of weighted harmonic sinusoidal oscillators to produce the a completely bandlimited sync signal but this would be prohibitively expensive for practical musical uses. One solution given in [4] proposed to implement hard sync by creating a bandlimited residual signal that is combined with the waveform at every reset instance. The reported results have been good but its implementation does incur a computational cost in identifying the instances of the reset before they occur as the residual is symmetric around the reset, a problem that also clearly occurs with the reset approach. Nam et al. [5] proposed to implement the hard sync by adding scaled versions of two bandlimited impulse trains and integrating the resulting signal.

2.2. Exact Fourier series for Hard Sync

An exact expression for the Fourier Series of the hard sync waveform can be obtained by viewing the waveform as the convolution of a pulse train with a rectangular windowed Slave sawtooth. The length of the window and the period of the impulse train are equal to the period of the Master. This can be written, where the asterisk denotes convolution, as

$$S_{sync}(t) = (S_{slave}(t)W(t)) * i(t) \tag{6}$$

where $W(t)$ is a rectangular window

$$W(t) = \begin{cases} 1, & 0 \le t < T_{master} \\ 0, & otherwise \end{cases} \tag{7}$$

where T_{master} is the period of the Master sawtooth, and $i(t)$ is the impulse train given by

$$i(t) = \sum_{k=-\infty}^{\infty} \delta(t - kT_{master}) \tag{8}$$

In the frequency domain, eq. (6) can be written

$$S_{sync}(\omega) = (S_{slave}(\omega) * W(\omega)) i(\omega) \tag{9}$$

The continuous-time fourier transform of the product of the sawtooth $S_{slave}(\omega)$ and the rectangular window $W(\omega)$ is

$$-j\frac{2}{T_{master}} e^{-j\frac{T_{master}}{2}\omega} \sum_{p\neq 0} \frac{1}{p} \mathrm{sinc}\left[(\omega_p - \omega)\frac{T_{master}}{2\pi} \right] e^{jp\pi(T_{master}/T_{slave})} \tag{10}$$

where T_{slave} is the period of the slave sawtooth, $\omega_p = p2\pi/T_{slave}$ and p is an index of sinc values.

The sinc in eqn. (10) stems from the continuous frequency spectrum of the rectangular window. The Fourier transform of a pulse-train of period T_{master} is itself a pulse train of frequency domain spacing $2\pi/T_{master}$. The time domain convolution of the windowed sawtooth with the pulse train is equivalent to the product of their spectra, eqn. (9), and this is the same as sampling of eqn. (10) in the frequency domain, which leads to an expression for the Fourier series of the hard sync wave

$$S_{sync}(k) = \frac{(-1)^k}{j\pi} \sum_{p\neq 0} \frac{1}{p} \mathrm{sinc}\left[\left(p\frac{T_{master}}{T_{slave}} - k \right) \right] e^{jp\pi(T_{master}/T_{slave})} \tag{11}$$

where k is the series index. Note that eqn. (11) describes the two-sided Fourier series.

Examining eqn. (11), it can be seen that every Fourier coefficient is made up of an infinite sum of sinc values indexed by the letter p. The infinite sum is due to the leakage resulting from the discrepancy between the zero-crossing rate of this sinc function in the frequency domain and the spacing of the sawtooth's harmonics. Figure 4 plots the set of Fourier series (upper panel) and the time domain signal (lower panel) using the same frequencies for the Master and Slave waveforms of Figure 3. The value of p is from -

20 to 20 and k is from -40 to 40, meaning 40 coefficients. The match between the waveform shapes in Figure 3 and 4 is evident. The only difference is that the one is Figure 4 is bandlimited. The spectral shape according to the Fourier series shows peaks whose frequencies are related to the frequency of Slave waveform. This links in with the following section about sync and its relationship with Comb filtering.

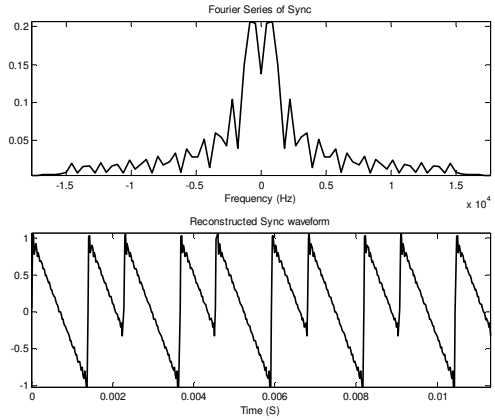

Figure 4: *Hard sync Fourier series (upper panel) and waveform (lower panel) computed using eqn. (11).*

2.3. Sync and its relationship to Comb filtering

Prompted by the fact that the appearance of the time-varying spectrum in Fig. 2 suggests that the effect is a form of comb filtering of the Master waveform, that is, the Hard sync of sawtooth waveforms can actually be written as the combination of weighted and phase shifted versions of the Master waveform. A suggestion to this effect was made already by [12] but not illustrated. The evidence for this is shown in Figure 4. If we consider the sync waveform in the lower panel of Figure 3, and subtract from it the Master waveform shown in the upper panel of Figure 3, we obtain the waveform in the upper panel of Figure 4. From the waveshape it can be hypothesized that it contains another sawtooth waveform at the same period of the Master but shifted in phase. This phase shift can be measured to be proportional to the time difference between the periods of the Master and Slave waveforms. Removing then a phase shifted master returns the waveform in the middle panel. This waveform is the combination of an impulse and a scaled and DC adjusted version of the Master waveform. Subtracting out such a waveform leaves a pure impulse train whose period is the same as that of the Master. This can be seen in the lower panel of Figure 4. An impulse train can be written as the differential of a sawtooth waveform. Thus, this hard sync waveform can be described as a combination of a number of Master sawtooths of different amplitudes, phase shifts and DC offset.

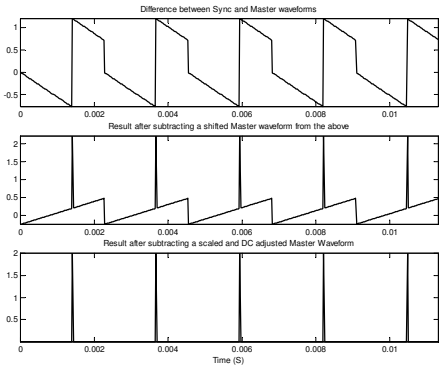

Figure 5: *Determining how the sync waveform is a combination of scaled, shifted and DC adjusted Master waveforms. The upper panel is the subtraction of the Master from the sync output, the middle panel shows what remains after subtracting a shifted Master, and the lower panel is the final remaining impulse train*

Further experimentation was carried out to verify that it always the case that the hard sync waveform can be formed from a set of Master waveforms. This phenomenon fits well with the Comb filtering interpretation of the sync effect as a Comb filter in essence combines a delayed and scaled version of the input with itself resulting in alteration of the waveshape and thus, the timbre of the input.

Through experimentation it was found that there were two key parameters: a weighting C and phase shifting ϕ that are both proportional to the period of the Slave waveform. The value for C is given by

$$C = \frac{\left(T_{master}/T_{slave} - \lfloor T_{master}/T_{slave} \rfloor\right)}{T_{slave}} \tag{12}$$

where T_{master} is the period of the Master sawtooth in samples, T_{slave} is the period of the Slave waveform in samples, and $\lfloor . \rfloor$ is the floor function. The actual Master waveform is weighted by this value C and is combined with a number of phase shifted versions of the Master. This value can be denoted as N and is computed by

$$N = \lfloor T_{master}/T_{slave} \rfloor \tag{13}$$

Thus, N phase shifted versions of the Master are required in the combination. Assuming an index n, with n=1…N, each version is phase shifted by an amount given in (14)

$$\phi_n = 2\pi n \frac{T_{slave}}{T_{master}} \tag{14}$$

The Hard sync waveform can then be written as

$$S_{sync}(t) = C\,S_{master}(t) + \sum_{n=1}^{N} S_{master}(t-\phi_n) \tag{15}$$

We write the Fourier series for a phase shifted, unity amplitude, falling Master sawtooth wave as

$$S_{saw}(2\pi f_{master} t + \phi_n) = \frac{2}{\pi}\sum_{k=1}^{\infty}\frac{\sin(2\pi k f_{master} t + k\phi_n)}{k} \tag{16}$$

with *k* being the harmonic index.

Then, the combination of sawtooths in the second term of the sync waveform in (15) can be expanded as a set of harmonically related sinusoids with *N* individual magnitudes and phases that are described using the absolute value and angle of their complex representation

$$\sum_{n=1}^{N} S_{saw}(2\pi f_{master} t + \phi_n) = \frac{2}{\pi}\sum_{k=1}^{\infty}\left|\sum_{n=1}^{N} e^{jk\phi_n}\right|\frac{\sin\left(2\pi k f_{master} t + \angle\left(\sum_{n=1}^{N} e^{jk\phi_n}\right)\right)}{k} \tag{17}$$

where $|\,|$ denotes the absolute value and $\angle$ is the phase angle.

If we then combine the first term of (15) with the expression in (17) we can get an expression for the sync wave as

$$S_{sync}(t) = 2\sum_{k=1}^{\infty}\left|C + \sum_{n=1}^{N} e^{jk\phi(n)}\right|\frac{\sin\left(2\pi k f_{master} t + \angle\left(C + \sum_{n=1}^{N} e^{jk\phi(n)}\right)\right)}{k\pi} \tag{18}$$

From (18) then, the magnitude of each harmonic in the spectrum of this Hard Sync wave form can easily be written

$$|H(k f_{master})| = \frac{2\left|C + \sum_{n=1}^{N} e^{jk\phi(n)}\right|}{k\pi} \tag{19}$$

Fig. 6 gives an example of a hard sync sync waveform and its Fourier series computed using eqns. (18) and (19) respectively. The frequency of the Master is 441Hz, the Slave is 1575Hz, and thus, after substitution into eqn. (13) the value of *N* is 3. The top panel displays the hard sync waveform and it is perfectly bandlimited, i.e., there is no aliasing distortion associated with it. The lower panel presents the Fourier series. Again, the resonances in the spectrum are related to the frequency of the Slave waveform.

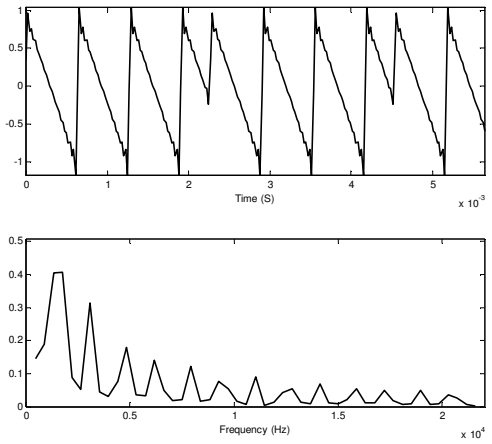

Figure. 6: *The upper panel shows a Hard sync waveform computed using eq. (18) and the lower panel gives its spectral magnitudes computed using eq. (19).*

3. IMPLEMENTATION USING A DELAY-LINE

Following the Additive Synthesis/Fourier series description of hard sync outlined in the previous section, it seems that it should also be possible to implement the phase shifting and scaling of the Master waveform via a delay-line filter approach. Previous work has already hinted at the potential for implementing hard sync using a time-varying comb filter [12]. However, no algorithm or expression has been provided for this method. Considering both the appearance of the spectrogram in Fig. 2, and the additive synthesis expression in (18) this appears to be a very appropriate model. The delay-line filter structure is an inverse comb filter. In the case where the period of the Slave is less than twice the period of the Master, it can be defined by

$$y(t) = Cx(t) + x(t-\tau) \tag{20}$$

where *C* is given by (1) and the delay-line length $\tau = T_{slave}$.

The transfer function of this filter is

$$H_c(z) = C + z^{-\tau} \tag{21}$$

The input to the comb filter is the Master waveform. In cases where the period of the Slave is two or more times than that of the Master, then additional comb filter stages must be cascaded. Fig. 7 shows a block diagram of the comb filter structure.

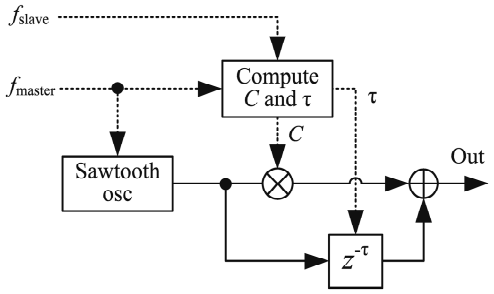

Figure. 7: *Block diagram of comb filter structure for generating hard sync waveform.*

In evaluation experiments carried out it was noticed that in cases where the frequency of the Slave is much higher than that of the Master it is necessary to have a DC blocker filter following the inverse comb filter structure. The final algorithm of those given in [13] was found to work well, where the value of the blocker filter delay was chosen to be half the period (in samples) of the input.

Fig. 8 gives an example of the output of the Delay-line filter approach where the frequency of the Master is again 441Hz, that of the Slave is 700Hz and the sampling frequency is 44100Hz. After substitution into eqn. (13) the value of N is 1. The input to the delay line filter was a bandlimited sawtooth. This sawtooth was generated using the algorithm given in [5] that creates it from a 3rd order B-spline bandlimited impulse response train. Fig. 8 plots a portion of the magnitude of the frequency response of the delay-line filter in the upper panel and its waveform output in the lower panel. As to be expected, the frequency response exhibits peaks at the harmonic frequencies of the Slave and nulls at frequency locations halfway in-between these. The waveform of the delay-line output in the lower panel clearly shows the Hard sync effect.

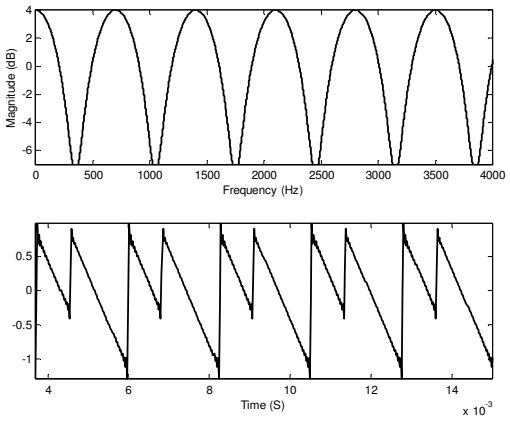

Figure. 8: *The upper panel shows a zoom of the low frequency portion of the frequency response of the delay-line filter and the lower panel shows the output sync waveform.*

For non-integer sample periods of the Slave waveform or to create a time-varying effect it is necessary to use fractional delays that can be implemented using techniques discussed in [14]. The distinct advantage of this delay-line filter-based approach is that if the filter input is bandlimited then the filter output will also be bandlimited. Additionally, it should be more computationally efficient than an approach based on the explicit phase reset approach, explained in section 2.1, as there is no decision logic required in its implementation.

4. EVALUATION

Firstly, to evaluate in relative, rather than absolute, terms the efficiency of the delay-line implementation a number of empirical evaluations were carried out to compare the execution time of the delay-line version against the algorithm presented in [5]. The algorithm in [5] implemented the phase reset hard sync, as discussed in section 2.1, applied to a bandlimited sawtooth. The author of [5] kindly gave a copy of his Matlab m-file for this. The input to the delay-line was also bandlimited sawtooth generated using his 3rd order B-spline algorithm of [5]. For the Hard sync evaluation the parameters in both cases was that Master frequency was held fixed at 441Hz and the Slave frequency was set to vary from $Fs/99$ to $Fs/3$ where Fs =44100Hz (the sampling frequency). These figures were chosen as it meant that the delay-line did not need any fractional delay elements included in its structure. The reason for this type of relative evaluation was that the aim was to show that the delay-line approach is very attractive. However, it is realised that the mechanics of any algorithm's implementation always has a significant impact on its efficiency in absolute terms. Thus, a proper evaluation of this, to do it justice, would require a separate study. The experiments were carried out on a Dell Latitude D620 laptop with an Intel Core 2 processor running at 1.66GHz.

Fig. 9 shows a plot of the average execution time taken over 20 simulations of the Hard sync algorithm in [5] against the delay-line approach. As can be seen from the plot, the delay-line approach is at least twice as fast. Furthermore, its execution time does not change significantly with respect to the Slave Frequency while this is not the case for the approach of [5] which actually increases as the relative Slave frequency increases. Thus, the delay-line approach is relatively faster and is consistently so. Note that for a real-time implementation, say within a VST software synthesizer, the actual absolute execution time of the delay-line approach could be reduced further using suitable code optimisation techniques.

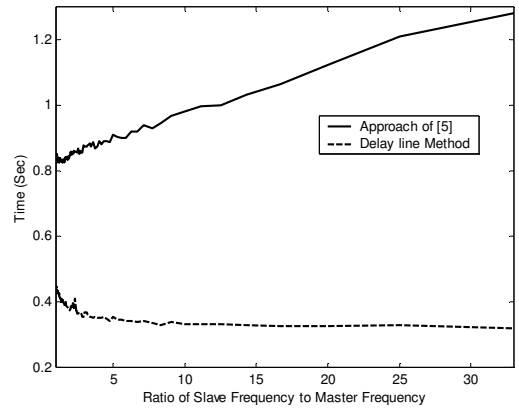

Figure 9: *A comparison of the execution time of Matlab m-files for to compute Hard-sync using the algorithm from [5] (solid line) and the delay-line approach (dashed line) for a fixed Master frequency and an increasing set of Slave frequencies.*

As a further test the difference between the Fourier series given by the exact expression, i.e. eqn. (11) in section 2.2, against the harmonic magnitudes from the spectrum of the output of the delay-line when the input is a bandlimited sawtooth was measured. Again, the parameters were that Master frequency was held fixed at 441Hz and the Slave frequency was set to vary from *Fs*/99 to *Fs*/10 where *Fs* =44100Hz (the sampling frequency).

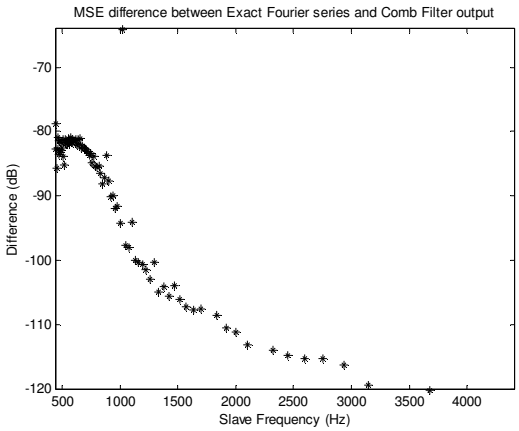

Figure 10: *A comparison of the MSE difference between the Hard Sync algorithm of [5] (solid line) and the delay-line hard sync (dashed line with triangles) with the Fourier series of eqn. (8) respectively for a fixed Master frequency and an increasing set of Slave frequencies.*

Fig. 10 plots the mean square error difference in dB between these. From the figure it can be seen that overall there is little difference between these versions of the sync signal, the error being less than -80dB in almost all cases. From this result, we can say that the exact expression and the delay-line approach produces, for all intents and purposes, the same output.

5. CONCLUSION

This paper has presented a Fourier Series description for Oscillator Hard Synchronisation that is associated with Analog subtractive synthesizers. This was followed by an alternative expression based on observations that showed how the sync waveform can be formed from scaled, shifted and DC adjusted versions of the Master waveform. The expressions then lead to a delay-line implementation for the Hard Sync operation. Comparisons with a recent alternative implementation showed that this version was relative faster computationally.

Future work will examine how the delay-line Hard sync algorithm can be emulated using distortion synthesis techniques such as VPS [15] and demonstrate its link with resonant synthesis as suggested in [16].

6. REFERENCES

[1] M. Russ, *Sound synthesis and sampling*, Focal press, Elsevier, Oxford, UK, 2009.

[2] Moog Music Inc., *Minimoog Voyager Old School user's manual*, 2008. Available: http://www.bigbriar.com/manuals/old_school_manual_1_0.pdf, [Accessed: Apr.1 2012].

[3] M. Jenkins, *Analog Synthesizers*, Focal press, Elsevier, Oxford, UK, 2007.

[4] E. Brandt, "Hard sync without aliasing," in *Proc. Int. Comp. Music Conf.,* Havana, Cuba, Sept. 17-22, 2001.

[5] J. Nam, V. Välimäki, J. S. Abel, and J. O. Smith, "Efficient antialiasing oscillator algorithms using low-order fractional delay filters," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 18(4), pp. 773–785, May 2010.

[6] J. Kleimola and V. Välimäki, "Reducing aliasing from synthetic audio signals using polynomial transition regions," *IEEE Signal Processing Letters*, vol. 19, no. 2, pp. 67–70, Feb. 2012.

[7] D. Yeh, *Digital implementation of musical distortion circuits by analysis and simulation*, Ph.D. thesis, Stanford University, Stanford, CA, USA, June 2009. https://ccrma.stanford.edu/~dtyeh/papers/pubs.htm [Accessed: Apr. 1. 2012].

[8] T. Helie, "Volterra Series and State Transformation for Real-Time Simulations of Audio Circuits Including Saturations: Application to the Moog Ladder Filter", *IEEE Trans. Speech and Audio Processing.*, vol. 18(4), May 2010, pp. 747-759.

[9] F. Fontana and M. Civolani, "Modeling the EMS VCS3 Voltage-Controlled Filter as a nonlinear filter network", *IEEE Trans. Speech and Audio Processing.*, vol. 18(4), May 2010, pp. 760-772.

[10] G. De Sanctis and A. Sarti, 'Virtual analog modeling in the wave digital domain,' *IEEE Trans. Speech and Audio Processing*, vol.18 (4), May 2010, pp. 715-727.

[11] G. Deslauriers and C. Leider, "A bandlimited oscillator by frequency-domain synthesis for virtual analog applications," *AES Convention 127,* New York, USA, Oct. 9-12, 2009.

[12] D. Lowenfels, "Virtual analog synthesis with a time-varying comb filter," *AES Convention 115,* New York, NY, USA, Oct. 10-13, 2003.

[13] R. Yates and R. Lyons, "DSP Tips & Tricks [DC Blocker Algorithms]", *IEEE Signal Processing Magazine*, vol. 25, no. 2, March 2008, pp. 132 – 134.

[14] T. Laakso, V. Välimäki, M. Karjalainen, and U. Laine, "Splitting the unit delay—Tools for fractional delay filter design," *IEEE Signal Processing Magazine*, vol. 13(1), Jan. 1996, pp. 30–60.

[15] J. Kleimola, V. Lazzarini, J. Timoney, and V. Välimäki, "Vector Phaseshaping synthesis," *Proc. of the 14th conf. on Digital Audio Effects (DAFX)*, Paris, France, Sept. 2011.

[16] M. Ishibashi, *Electronic Musical Instrument*, U.S. Patent no. 4,658,691, 1987.