A SYNTHESIZER BASED ON SQUARE WAVES

Sossio Vergara
Universidad ORT,
Montevideo Uruguay

ABSTRACT

One of the most widely employed technique for the sound synthesis is based on the Fourier theorem that states that any signal can be obtained as a sum of sinusoids. Unfortunately this algorithm, when applied to synthesizers, requires some peculiar operations, as the addressing of a Look Up Table, that are not easily built-in in standard processors, thus requiring specially designed architectures.

The aim of this paper is to show that, when using a new method for the analysis and polar coordinates, a much broader class of functions can be employed as a basis, and it turns out that the square wave is one of such functions. When the synthesis of signals is carried out by summing square waves, the additive synthesizer architecture results much more simplified, allowing for example to synthesize complex sounds simply in software, using general purpose microprocessors, even in real-time.

Firstly it will be proven that the L^2 function space admits a broad class of functions as a basis, and the requirements for a function, in order to be a basis, will be defined. A straightforward and computationally simple algorithm for the analysis of any function on such generic basis will be proposed. Finally the architecture for the square wave based synthesizer will be sketched and examples of synthesized waveforms will be given.

1. INTRODUCTION

The Fourier theorem has been around for exactly 200 years and has evolved a lot in this time; from the original series a whole set of tools has been derived.

The Theorem in its standard form states that any function (and hence any real world signal) that is periodic, has finite energy and a limited number of discontinuities in a cycle, can be decomposed as a series of sine and cosine. Or, better, that the series produced by the superposition of the sinusoids, are said to converge to the function. Actually some different definitions of convergence exist: pointwise, uniform and L^2-norm, the last being the most satisfactory from the mathematical point of view.

By definition the norm of a function *f(x),* periodic with a period P, is:

$$\| f(x) \| = \left[\frac{1}{P} \int_0^P | f(x) |^2 \, dx \right]^{\frac{1}{2}} \quad (1)$$

And the series of partial sum S_N (basis superposition) is said to converge to *f(x)* in norm if:

$$\lim_{N \to \infty} \frac{1}{P} \int_0^P | f(x) - S_N(x) |^2 \, dx = 0 \quad (2)$$

Up to now we did not specify how the series S_N shall be built, the only thing that matters is that it can be built from a specific function or couple of functions, called the basis, and a set of coefficients that can be computed univocally for any function *f(x).*

As a matter of fact the Fourier theorem has a rectangular and polar enunciation and is valid also for other orthogonal bases as the Legendre polynomials.

The standard enunciation of the Fourier Theorem states that:

Given a function *f(x)*, periodic with period P = 2π, that satisfies the Dirichlet conditions:

1) *f(x)* has a finite number of maxima and minima in one period.
2) *f(x)* has a finite number of discontinuities in one period.
3) and: $\int_0^P |f(x)| dx < \infty$

Then, *f(x)* can be decomposed in a series of sine and cosine as:

$$f(x) = C_0 + \sum_{k=1}^{\infty} \left(a_k \sin kx + b_k \cos kx \right) \quad (3)$$

Where:

$$C_0 = \frac{1}{2\pi} \int_{-\pi}^{+\pi} f(x) dx \quad (4)$$

$$b_k = \frac{1}{\pi} \int_{-\pi}^{+\pi} f(x) \cos kx dx \quad (5)$$

$$a_k = \frac{1}{\pi} \int_{-\pi}^{+\pi} f(x) \sin kx dx \quad (6)$$

Or, in the more compact, complex exponential form:

$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{jkx} \quad (7)$$

Where:

$$c_k = \frac{1}{2\pi} \int_{-\pi}^{+\pi} f(x) e^{-jkx} dx \quad (8)$$

The above equation is called the Fourier Series; it can be extrapolated to continuous and aperiodic functions by means of the Fourier Transform (and Discrete Fourier Transform in the case of Discrete-Time signals) and its inverse:

$$T_F(f(x)) = F(\omega) = \int_{-\infty}^{+\infty} f(x) e^{-j\omega x} dx \qquad (9)$$

$$f(x) = T_F^{-1}(F(\omega)) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{j\omega x} dx \qquad (10)$$

The sufficient condition for the existence of the Fourier Transform is that $f(x)$ be (Lebesgue) square integrable:

$$\int_{-\infty}^{+\infty} |f(x)|^2 dx < \infty$$

or equivalently $f(x) \in L^2$.

The Eq.:3 is a rectangular or Cartesian decomposition, we can think to the cosine as the horizontal axis and the sine its perpendicular. There is an alternative form of the (3) in terms of polar coordinates using the module b_k and phase φ_k:

$$f(x) = C_0 + \sum_{k=1}^{\infty} b_k \cos(kx + \varphi_k) \qquad (11)$$

And an equivalent for the transform [9].

The Fourier transform always works on the entire duration of the signal from time -∞, to +∞ , then any time information is spread out on the entire frequency axis.

To overcome this limitation another mathematical tool has been developed: the Short Time Fourier Transform (STFT); here the analysis is performed not on the entire signal but on chunks of a certain duration, as to be considered stationary in that time frame. Thus extracting both frequency and time information, i.e. the evolution of the signal. This kind of tools are said "time–frequency" analysis.

In the last decades a new mathematical tool has evolved: the wavelets [1][2][7]; these, initially developed as orthogonal functions were then generalized to the non-orthogonal case. They have the advantage of changing the window duration with the frequency. So, while in the STFT the analysis window has a fixed duration at all frequencies (it is a "fixed resolution analysis"), in the wavelets the analysis window is small at high frequency (better time resolution) and larger at low frequencies (better frequency resolution): so it is a "multiresolution" analysis.

Unfortunately the wavelets have some disadvantages in the reconstruction phase; only a limited set of wavelets is available and they all are special mathematical functions [1], [6].

In the last years a new concept has evolved for the time-frequency decomposition: the Frames [1], [3], [4]. The frame theory establish the general conditions under which one can recover a vector f in a Hilbert space **H** from its inner products with a family of vectors $\{f_n\}_{n\in\Gamma}$ called the "frame". Having such a general definition both, the discrete windowed Fourier transforms and the discrete wavelet transforms (both based on inner product) can be considered as special cases of frames and studied accordingly. The main advantage of the frames is that they can be generalized to being redundant, or overcomplete, which allows the reconstruction of signals even in presence of noise.

But the frame theory has some limitations: to reconstruct a signal from its inner products one needs to invert the frame operator U:

$$U f[n] = \langle f, f_n \rangle \qquad \forall\, n \in \Gamma, \text{ an index set}$$

The inversion of this operator is always numerically complex and in some cases can be carried out only employing a dual frame (a reconstruction frame different from the analysis frame). Due to these difficulties only few special types of frames are actually employed, and mainly in the transmission and reconstruction of signals in noisy environment.

Lately, a new approach to the function decomposition has been proposed: the Generic Frequency Analysis, [10], [11], [12]. There, it has been demonstrated that any function f in L^2 can be reconstructed by a series of couples of even and odd functions that have multiplicative Fourier coefficients [11].

Indeed, having two functions like the even and odd:

$$X(x) = \sum_{n=1}^{\infty} A(n)\cos(nx) \qquad (12)$$

$$Y(x) = \sum_{n=1}^{\infty} B(n)\sin(nx) \qquad (13)$$

That have quadratically summable and completely multiplicative Fourier coefficients, i.e.:

$$\sum_{n=1}^{\infty} A^2(n) < \infty$$

and:

A(nm)=A(n)A(m)

and the same for B(n).

Then, the combinative system:

1, X(x), Y(x), X(2x), Y(2x), …, X(nx), Y(nx) is a basis in the whole space $L^2[-\pi, \pi]$.

It means that any function $f(x) \in L^2[-\pi, \pi]$ can be expressed as a series of these functions as:

$$f(x) = C_0 + \sum_{n=1}^{\infty} C(n)X(nx) + D(n)Y(nx) \qquad (14)$$

With the coefficients given by:

$$C_0 = \int_{-\pi}^{\pi} f(x) dx \qquad (15)$$

$$C(n) = \int_{-\pi}^{\pi} f(x) h_n(x) dx \qquad (16)$$

$$D(n) = \int_{-\pi}^{\pi} f(x) g_n(x) dx \qquad (17)$$

Here the functions:

$$h_n(x) = \sum_{d|n} A^{-1}\left(\frac{n}{d}\right)\cos(dx) \qquad (18)$$

$$g_n(x) = \sum_{d|n} B^{-1}\left(\frac{n}{d}\right)\sin(dx) \qquad (19)$$

Are the biorthogonal functions of X(x) and Y(x) respectively, where $d|n$ means that d is a factor of n. Details can be found in [10], [11], [12].

Summarizing, this last analysis/synthesis procedure starts by choosing a suitable couple of functions X(x) and Y(x); from these, the family of their biorthogonal functions: $h_n(x)$, $g_n(x)$ is generated. Finally the coefficients are calculated via the inner product of the function *f(x)* with any of these $h_n(x)$, $g_n(x)$.

Here the main restriction seems to be the necessity of using the inner product in the computing of the coefficients C(n) and D(n); this, in turn, forces to the use of biorthogonal functions and limits the set of functions that can be used as bases to even and odd couples of square waves and few others.

These results are quite recent and did not find many applications yet, although in [10] it was suggested a schematic for generating, with an analog circuitry, any *f(t)* as a sum of many even and odd square waves.

Another problem here lays in the complexity of the calculation of the coefficients C(n) and D(n) by means of the biorthogonal functions.

All the previous tools actually limit the set of possible basis to few functions; the same frame theory, that theoretically admits an ample class of basis functions, at the same time suffers from a great complexity in the computation, so limiting its applicability.

We now try a new approach to functional analysis that has the advantage of allowing a much more ample class of functions as basis while at the same time simplifying the computation.

All we have to do is not limit ourselves to the use of the inner product in the coefficient computation [8]. The new methodology being so general that it can be applied to the decomposition of a function in terms of both rectangular or polar coordinates. Only this last case will be discussed here, while for a general discussion and a deeper look at the consequence and some other application see [8].

We first need to find out the necessary requirements for a function S(x) in order to be a basis in L^2, when using polar coordinates reconstruction. We will make use of the norm convergence to prove that if S(x) satisfy some loose requirements, unique sets of modules M_n and phases Θ_n exist, such that the series S_N converges to any function. So that we can write:

$$f(x) = C_0 + \lim_{N\to\infty}\sum_{k=1}^{N} M_k S(kx+\Theta_k) \qquad (20)$$

Once we prove that these coefficients (the module and phase) exist and are unique, we will look for a way to find them. A last thing to note is that we will base the proof and the analysis, entirely on the reconstruction algorithm, i.e. the reconstruction algorithm will be used to create a function that has the same Fourier components as the original, and hence thanks to the uniqueness of the Fourier transform, it is actually the same function.

2. PROOF OF FEASIBILITY OF THE RECONSTRUCTION

Given a periodic function *f(x)* that can be expressed as a Fourier series in polar coordinates:

$$f(x) = \sum\nolimits_k \left[m_k \cos(kx+\vartheta_k)\right] \qquad (21)$$

and given another nonzero periodic function with Fourier series:

$$S(x) = \sum\nolimits_p \left[\ s_p \cos(px+\phi_p)\ \right] \qquad (22)$$

with zero average over a period, if the energy of *S(x)* is mainly at the fundamental frequency i.e.:

$$(\mathbf{I_p}) \quad |s_1|^2 > \sum_{i=2}^{\infty} |s_i|^2 \qquad (23)$$

Then *f(x)* can also be expressed as a series of *S(x)* in polar coordinates :

$$f(x) = \sum_{n=1}^{\infty}\left[M_n S(nx+\Theta_n)\right] \qquad (24)$$

Here the functions are assumed to be periodic in [0,2π]; however it is easy to extend this period P to any other period P' just changing from variable x to variable z = xP'/P.

Moreover, without any loss of generality, we assume that the average, or DC component of the function *f(x)* is zero, otherwise a constant C_0 should be added. But this will not change our conclusions. We split the proof in two parts, in the first part we will prove that any function S(x) $\in L^2$ is complete on that space; the second part defines the requirements for such S(x) in order to be a basis, i.e. under which conditions the series of the S(kx) converges (in norm) to any *f(x)* $\in L^2$.

2.1. Existence and uniqueness of the coefficients

From Eq.: (21) and (24) we can write:

$$\begin{aligned} f(x) &= \sum\nolimits_k \left[m_k \cos(kx+\vartheta_k)\right] = \\ &= \sum\nolimits_n M_n \sum\nolimits_p \left\{s_p \cos\left[p(nx+\Theta_n)+\phi_p\right]\right\} \end{aligned} \qquad (25)$$

As the cosine is an orthogonal function we can separate the components of *f(x)* at each frequency: F1, F2, , etc.:

F1: $$m_1\cos(x+\vartheta_1) = M_1 s_1 \cos\left(1(x+\Theta_1)+\phi_1\right)$$

F2:

$$\begin{aligned} m_2\cos(2x+\vartheta_2) &= M_1 s_2 \cos\left(2(x+\Theta_1)+\phi_2\right) + \\ &+ M_2 s_1 \cos\left(1(2x+\Theta_2)+\phi_1\right) \end{aligned}$$

Etc.

At the fundamental frequency F1 we can separate the variables and find the unique solutions:

$$M_1 = \frac{m_1}{s_1} \tag{26}$$

$$\Theta_1 = \vartheta_1 - \phi_1 \tag{27}$$

We can then place these two values in the equation for the frequency F2 and find the new M_2 and Θ_2; iterating the procedure we can reconstruct the exact module and phase of the function *f(x)* at any frequency by means of a series of the functions S(nx).

In other words, we just require that the *f(x)* and its reconstructed counterpart (as a sum of S(x)) have the same Fourier spectrum. We obviously can do it at the fundamental F1 (Eqs. 26 and 27) then, we do the analysis starting from frequency 1 up, every time subtracting the previous reconstruction from the function *f(x)*, in order to find the next resulting components. This analysis-reconstruction-analysis procedure can be stopped whenever the difference, i.e. the RMS error, in reconstruction be lower than a predetermined value.

For example if we reconstruct a square wave using a square wave as basis, only one component will be needed, as it zeroes the (Fourier) harmonics of the difference all at once. Intuitively, having any function S(x) we can modify its module and phase in order to match any possible module and phase at frequency 1; then using S(2x) we can do the same for frequency 2 and so on.

2.2. Convergence of the series

In the previous paragraph we actually proved that any function can be used to reconstruct, by means of the given algorithm, any other function at any single frequency. Mathematically speaking we showed that **any** nonzero function spans the entire L^2 space when employing the given analysis/reconstruction (we could say 'de-construction") method. Or equivalently, the function S(x) is complete on the space L^2.

Now, to check whether S(x) is a basis, we need to verify under which conditions the series of the S(kx) **converges** to any *f(x)*. We can express it in more rigorous terms requiring that the norm of the error tends toward zero as N increases (following the definition of metrics in Hilbert spaces):

$$\lim_{N\to\infty} \| f_{eN}(x) \| = \lim_{N\to\infty} \| f(x) - \sum_{n=1}^{N} M_n S(nx+\Theta_n) \| = 0 \tag{28}$$

As the norm is always positive, and we eliminate one Fourier component at any iteration, we can reduce the above problem to the search of requirements on the function S(x) that verify:

$$\|f_{eN1}\| > \|f_{eN2}\| \quad \forall \ N1 < N2 \tag{29}$$

We can start from zero, or, no reconstruction at all, then the error is the function itself:

$$\|f_{e0}(x)\| = \|f(x)\| \tag{30}$$

We require that the first approximation has an error lower than this:

$$\|f(x)\| = \|f_{e0}(x)\| > \|f_{e1}(x)\| \tag{31}$$

Using Eq.: 21 and 22

$$\begin{aligned} \| f_{e1}(x) \| &= \| f(x) - M_1 S(x+\Theta_1) \| = \\ &= \| \sum_k m_k \cos(kx+\vartheta_k) - \\ &- M_1 \sum_p s_p \cos(px+\phi_p+\Theta_1) \| \end{aligned} \tag{32}$$

We can rewrite the right hand side of Eq.:32 as:

$$\begin{aligned} &\| m_1 \cos(x+\vartheta_1) - M_1 s_1 \cos(x+\phi_1+\Theta_1) + \\ &+ \sum_{k\geq 2} m_k \cos(kx+\vartheta_k) - \\ &- M_1 \sum_{p\geq 2} s_p \cos(px+\phi_p+\Theta_1) \| \end{aligned} \tag{33}$$

But, as M_1 and Θ_1 are the results of Eqs.: 26 and 27, the sum of the first two terms is zero, then:

$$\| f_{e1}(x) \| = \| \sum_{k=2}^{\infty} m_k \cos(kx+\vartheta_k) - M_1 \sum_{p=2}^{\infty} s_p \cos(px+\phi_p+\Theta_1) \| \tag{34}$$

While from the Eqs. (26) and (27) we can rewrite *f(x)* as:

$$\begin{aligned} \| f(x) \| &= \| m_1 \cos(x+\vartheta_1) + \sum_{k=2}^{\infty} m_k \cos(kx+\vartheta_k) \| = \\ &= \| \sum_{k=2}^{\infty} m_k \cos(kx+\vartheta_k) + M_1 s_1 \cos(x+\Theta_1+\phi_1) \| \end{aligned} \tag{35}$$

Comparing the second part of Eq.:35 and Eq.:34 we can see that $\|f_{e1}(x)\| < \|f(x)\|$ is verified if:

$$|s_1|^2 > \sum_{p=2}^{\infty} |s_p|^2$$

That is the condition of convergence $(\mathbf{I_p})$ above.

We can now iterate the procedure, using each time as the new function *f(x)* the error function found at the previous stage; and, as the limit of the Fourier coefficients m_k is zero (Eq.: 21), and the norm is always positive, the error in norm shall tend to zero.

Then we proved that any function, or signal, *f(x)*, that can be decomposed in Fourier series, can also be uniquely decomposed in series of any other function S(x) satisfying the requirement $\mathbf{I_p}$ above; and hence S(x) is a basis (of the L^2 Hilbert space) when using the above "deconstruction" algorithm and polar coordinates.

This result can be expressed in a more intuitive way if we remember that the energy of a signal is proportional to the sum of the squares of its Fourier coefficients; then the condition $\mathbf{I_p}$ dictates that most of the energy must be concentrated at the fundamental frequency of the basis function S(x). So anytime we make an approximation as the $\|f_{e1}(x)\|$ in the Eq.: 32, we subtract more energy at the fundamental frequency than we add at higher frequencies. Then, iterating the procedure the total energy of the difference (the norm of the error function) approaches zero.

Once we make sure that with the right coefficients we can assemble the Eq.: 20, all it remains to do is to find the coefficients M_n and Θ_n. We can simply use the same algorithm we employed in the proof above, starting from frequency 1 and subtracting at any frequency the approximation found at the previous step.

The procedure is very efficient, as it is essentially a recursive change of coordinates between the sinusoids and the new basis S(x). An even faster algorithm that works directly in the frequency domain has been written, so taking advantage of the speed of the FFT.

Note how this tool, thanks to its recursive algorithm has considerable advantages over above mentioned wavelets, frames and generic function analysis.

3. CONSEQUENCES OF THE NEW DECOMPOSITION

The fact that we found that such an ample class of functions or even real world signals, are basis of the L^2 space has some unexpected consequences, some of them are pointed out in [8].

Here we will focus on the advantages that such approach can have on the sound synthesis. Now that we know how to decompose any sound, not just as a sum of sinusoids, but as a sum of much more complex functions, we can generate richer sounds adding only few instances of a complex basis, instead of many simple sinusoids. Indeed, it is much more efficient to synthesize a complex sound starting from a basis that has some similarity with it.

The consequence on the computing power needs is evident. It is like having the computational efficiency of FM generation but with the control detail of the additive synthesis.

Or one can just imagine the possibilities of synthesize a complex sound like a piano, but adding samples taken from a different sound, for example the human voice.

But we explore now the opposite direction: we will try to find a way to simplify the additive synthesizer, so that a much cheaper system could be feasible, without sacrificing any of the features of the standard (Fourier) additive synthesizer.

The trick is as simple as using a special function like the square wave as a basis. Indeed the 2 level (+1, -1) square wave, satisfies the requirement I_p above, so it is a basis of the L^2 space, including any real world signals too.

Just to prove it in Fig. 1 there is the plot of a sinusoid obtained as a sum of 21 square waves.

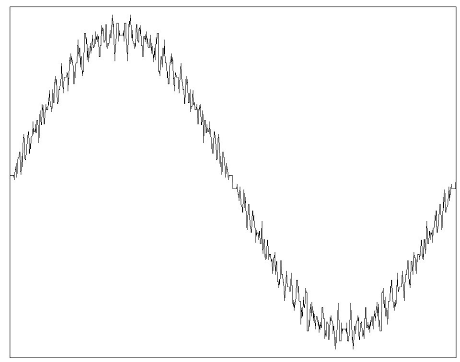

Figure 1:The sinusoid as a sum of 21 square waves

If the sinusoid seems too elemental to prove it, following there is a more complex signal obtained as a sum of square waves.

In Fig.2 a complex signal has been approximated with just 9 square waves. If we consider that the original signal is the sum of 7 Fourier components (sinusoids), and extends up to a frequency 11 times the fundamental, the approximation is surprisingly good. In Fig. 3 the same signal has been better approximated by summing 36 square waves, up to a frequency that is 50 times the fundamental. Always in the figures dotted line represent the signal while solid line is its reconstruction. Just as a reference in the Fig. 4 there is the "square wave frequency spectrum" i.e. is the plot of the module M_k of the square waves as a function of frequency, relative to the approximation of Fig. 3.

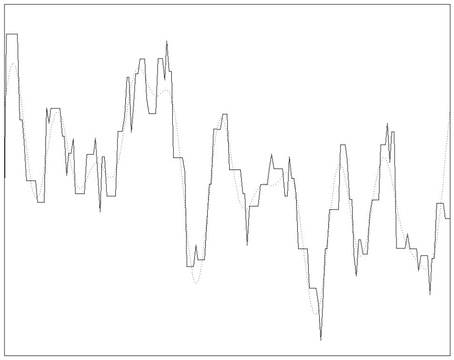

Figure 2: A complex signal as a sum of 9 square waves

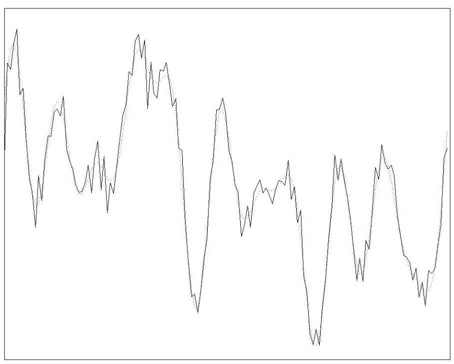

Figure 3: The same complex signal obtained as a sum of 36 square waves

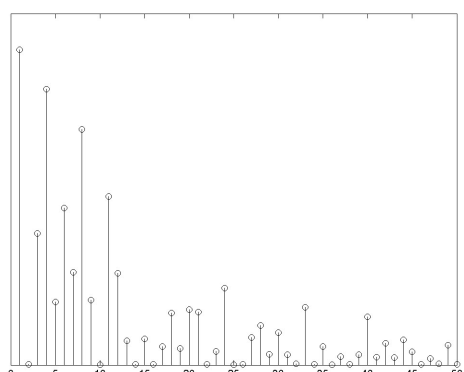

Figure 4: The "square wave spectrum" relative to the analysis of the signal in Figure 3

A corresponding plot of the phases could be done. Note that here we refer to phases or delays, while leaving the term "angles" only to the trigonometric fun ctions.

One could even think to modify the "generic spectrum" as the one in Fig. 4 to get some kind of "generic filtering"; well, the results on the reconstructed signals would be absolutely counter-intuitive but nonetheless interesting.

About the high frequency noise that appears in the previous figures, it should not worry as it is easily eliminated by a simple standard RC low-pass filter. That noise is, in some way, the opposite of the Gibbs phenomenon encountered at the discontinuities of a square wave obtained as a sum of sinusoids. As the sinusoid is a continuous function, the Fourier series cannot pointwise approximate discontinuities as those found in square waves. Conversely, as the square wave is a function with discontinuities, a series of them cannot pointwise approximate a continuous function as is the sinusoid, but for our purposes the norm approximation is good enough as far as we can eliminate the high frequency noise with a plain Low Pass filter.

Some points should be emphasized: when "de constructing" a sinusoid with square waves, the series extends to infinite (and the same when approximating a square wave with sinusoids). In Figure 1 the square wave spectrum is extended up to 50 times the fundamental, while in Figure 3 the highest square wave frequency is only 3 times the highest Fourier component in the signal. But we must consider that, at the Nyquist frequency (half the sampling rate), the sinusoid and the square wave are both sampled with 2 points only, so they are indistinguishable and hence, essentially, the same wave.

At this point one could ask why do not use the wavelets for the same purpose, as for example the Haar set that is obtained from a square wave as in Fig 5.

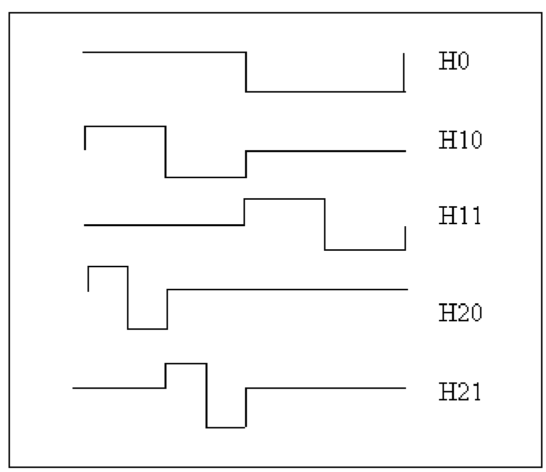

Figure 5: Example of Haar Wavelets

The reason is that the hardware results a little more cumbersome as these are actually three level waves (+1, -

1 and 0), and much more wavelets are required for the reconstruction. For example in Fig. 6 a sinusoid has been obtained as a sum of 32 wavelets; it can be seen that the approximation is worse than in the previous Fig. 1, where only 21 square waves were used.

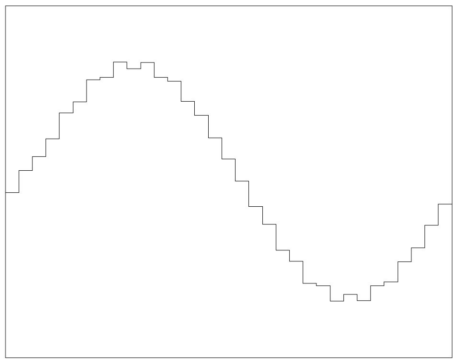

Figure 6: The sinusoid obtained as a sum of 32 Haar wavelets

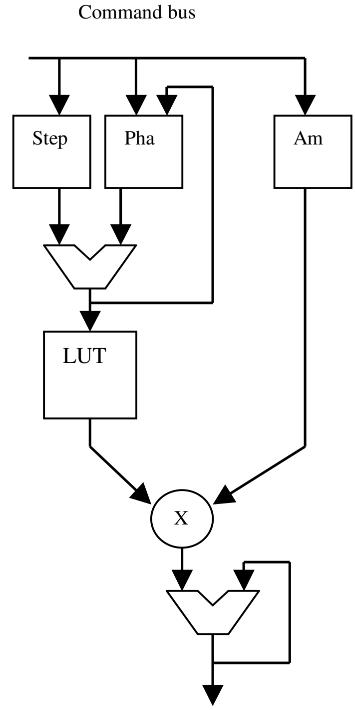

Figure 7: Additive Fourier synthesizer

4. ARCHITECTURE OF THE SQUARE WAVE ADDITIVE SYNTHESIZER

To see what the architecture of a square wave based synthesizer would be, we first look at a typical standard additive synthesizer based on Fourier synthesis as the one in Fig. 7. As it is sinusoid based, it needs a way to efficiently compute the sinusoids, the typical solution is to introduce in the architecture a Look Up Table (LUT) and eventually a mechanism for the interpolation of samples not contained in the LUT.

Once a sinusoid sample has been computed it must be multiplied by a given amplitude, that is essentially the module in the Fourier series expansion for that signal (stored in the Am memory).

Finally all these must be accumulated to produce a single sample of the synthesized signal. Note that we do not included an envelope controller at the output as it would be just a low rate multiplication.

Now let's simplify the previous architecture to accommodate the square wave basis. We do not need the LUT as the square waves are purely digital, and, as we associate the two levels of the square wave at the values of +1 and –1 we do not even need the multiplier, a simplest 2's complement logic will be sufficient. So the architecture for the square wave based synthesizer would be as in Fig. 8. The square wave is simply the most significant bit (msb) of the phase, it controls a 2's complement logic that is the equivalent to a multiplication by 1 or –1 of the module amplitude (Am).

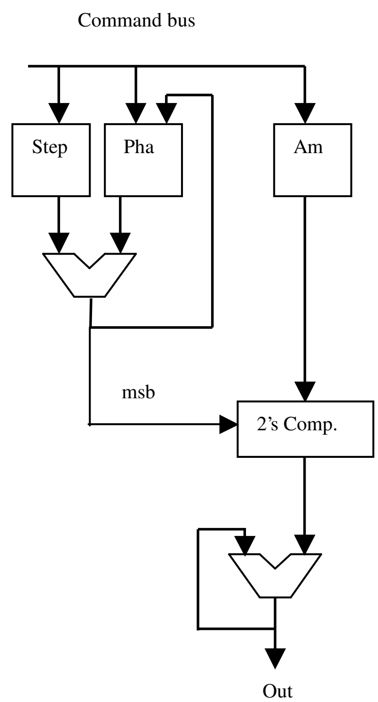

Figure 8: Additive square wave synthesizer

We can see that only two adders and some memory space are needed in order to synthesize any signal. It means that even very cheap devices can generate complex sounds. Looking at high end application like music synthesis, one can see that, in order to reduce the high frequency noise associated with square wave synthesis as in figures 2 and 3, one could include more square waves at higher frequency. Hence one can establish a sampling rate higher than that required for the Fourier synthesis, and so easing the job of the final low pass filter (it's essentially a kind of oversampling).

The reason is that the Nyquist-Shannon theorem is actually valid only for sinusoidal decomposition, but can be generalized to accommodate for any basis decomposition.

Just to make an example, let's think to the decomposition of a square wave in the sinusoid basis, then infinite sinusoids would be necessary, as the frequency spectrum of the square wave extends to infinite.

While, when analyzing a sinusoid, only one Fourier component is needed, so the Nyquist frequency in that case would be just the double of the signal frequency. The opposite is true when the basis is the square wave; the square wave spectrum of a sinusoid is infinite, while that of a square wave is a single frequency.

As in music we mostly deal with harmonic sounds, they are more similar to sinusoids than to square waves, but, as said above, a simple low pass filter at the output cuts all the unwanted noise. For instance, an ideal (Fourier) low pass filter at half the Nyquist frequency would eliminate all higher Fourier components so, for example reducing a square wave at the cut-off frequency to a plain sinusoid. As ideal filters do not exist in the real world, all one has to do is to double the sampling rate of the signal generated by square waves, and use a real low pass filter. It is the equivalent to reconstructing the highest frequency sinusoid as a sum of two square waves. This is enough to reproduce high quality sounds.

This leads us to a further enhancement to the architecture of Fig. 8; there the computation is carried out at any cycle, i.e. to generate any sample of the sound all the square wave components must be added. Let's suppose we are generating 2 channels at 100KHz of sampling rate, then we have 5 microsecond of computing time for each channel. If we have 100 square wave oscillators for each channel, it means that we have 50 nanoseconds cycle time for each square wave oscillator. Almost any modern microprocessor can do the job.

But we can further enhance it.

The fact is that only the highest frequency square wave changes at any cycle, the lower frequency square waves remain constant for many of the clock cycles. It is an additional advantage respect to the sine wave additive synthesizer. For example in Fig. 9 a sinusoid is plotted together with its first few square wave components. It can be seen that for most of the time only few of the square waves change sign contemporarily; while the rest remaining constant, so it is useless to sum them at each cycle. Then a sort of differential technique can be used; the output is kept constant and only when one of the square wave switches, the output is updated for the relative factor.

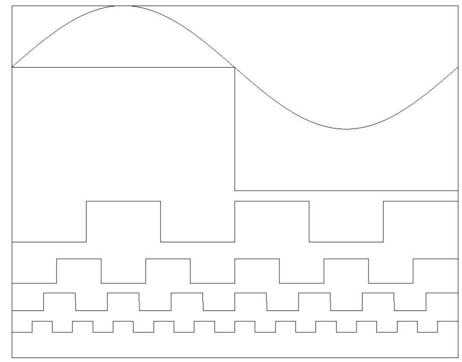

Figure 9: The sinusoid and its first square wave components

This approach is best suited for software implementations, in which a processor can time share music synthesis and other jobs. An even simpler version, when using only few square wave oscillators, can be designed using just an adder, few counters and some registers.

It can be seen that a synthesizer based on square waves is a viable alternative in many different applications, as a matter of fact it has the advantage respect to the sinusoid additive synthesizer of allowing many possible implementations.

When very high frequency signals are needed, a completely custom integrated circuit is viable; in this case the sampling rate can be foreseen in the order of many tens or even hundreds of MHz, so allowing for digital synthesis in a field where the only viable implementations were, up to now, analog.

Instead, for signals in the audio band, a dedicated DSP processor or even one of the new general purpose microprocessors can be employed, thus greatly reducing the cost of such instruments, while retaining the effectiveness of the additive synthesis.

This approach can thus effectively lead to software only, real-time synthesis on standard personal computers.

Finally, very simple, low end implementations are possible, so allowing this technique to be promptly used in a large class of devices. An experimental device has already been created by students at the ORT University in Montevideo using just a simple FPGA. It has shown satisfactory audio capabilities, needing only a DAC and a plain RC filter at the output in order to generate low noise signals. Other implementations of the technique are actually under study.

5. AKNOWLEDGMENTS

The author wishes to thank prof. S. Cavaliere who introduced him to the computer music, for his constant support and friendship.

6. REFERENCES

[1] Daubechies I. *The wavelet transform, time-frequency localization and signal analysis*. IEEE Transactions on Information Theory, 36(5):961-1005, 1990.

[2] Daubechies, I. *Ten Lectures on Wavelets*. SIAM, Philadelphia. 1992

[3] Daubechies, I. *From the original framer to present-day time-frequency and time-scale frames*. The J. Fourier Anal. Appl. 3 1997.

[4] Duffin, R. J., Schaeffer, A. C. *A class of nonharmonic Fourier series*. Trans. Amer. Math. Soc. Vol. 72 1952.

[5] Haar A. *Zur Theorie der orthogonalen Funktionensysteme*. Mathematische Annalen, 69:331-371, 1910.

[6] Koc, C. K., Chen, G., Chuy, C. K. *Analysis of computational methods for wavelet signal decomposition and reconstruction*. IEEE Transaction on Aerospace and Electronic Systems. July 1994

[7] Mallat, S. G., .*A theory for multiresolution signal decomposition: the wavelet representation*. IEEE Transactions on Pattern Recognition and Machine Intelligence. Vol. 11, No. 7, pp. 674-693, July 1989.

[8] Vergara, S., *On generic frequency decomposition.* To be published.

[9] Walker, J.S. (1988). *Fourier Analysis.* Oxford Univ. Press, Oxford.

[10] Wei, Y., Chen, N.: *Square wave analysis*. Journal of Mathematical Physics Vol. 39, N. 8, August 1998.

[11] Wei,Y., *Frequency analysis based on general periodic functions*. Journal of Mathematical Physics Vol 40 N. 7, July 1999

[12] Wei, Y.: *Frequency analysis based on easily generated functions*. Applied and Computational Harmonic Analysis, 2000