AUDIO SYNTHESIS BY BITWISE LOGICAL MODULATION

Jari Kleimola

Helsinki Institute for Information Technology (HIIT)
Helsinki University of Technology, TKK
Espoo, Finland
jari.kleimola@hiit.fi

ABSTRACT

The synthesis of rich audio spectra requires usually complex source waveforms, a large number of simple source components, or increased algorithmic complexity. This paper describes an implementation, which shows that simple elementary bitwise logical operations (OR, AND, XOR) possess power to produce such spectra. Applying these operations to two sinusoidal audio oscillators produced wide variety of new harmonically related sonic material. The synthesis method is efficient to implement and easily controllable, but it is not generally band-limited.

1. INTRODUCTION

A sonically pleasing waveform is often associated with rich and evolving spectrum. To produce such sounds, some well-known synthesis techniques use simple arithmetic operations to fuse elementary waveforms, like sinusoids, together. For example in additive synthesis, sinusoidal partials are *added* together to form a more complex spectral composition, while amplitude (AM) and ring modulation (RM) use digital domain *multiplication* between the source signals to strive towards the same goal [2] (cf. fig. 1a). One of the problems with these approaches is that, in order to generate a rich spectrum, there has to be a large number of source components, each with a separate set of control parameters (cf. figs. 1b-c). Alternatively, one has to use more complex source waveforms, but this reduces the amount of control, possibly also resulting inharmonically related spectra. Frequency modulation [3] uses sinusoidal sources, but requires more computation effort than basic arithmetic operations.

Addition and multiplication are not the only elementary operations that can be performed between audio rate signals, though. In fact, some early analog synthesizers used square waves and exclusive-or (XOR) operation to produce sounds similar to those generated by ring modulation [4]. This paper shows that when bitwise logical operations are applied to sinusoidal source waveforms, it is possible to generate rich and evolving spectra (cf. fig. 1d). The generated spectrum can be easily mutated and animated using a set of familiar control parameters for oscillator amplitude, frequency and phase offset.

The structure of this paper is as follows. Section 2 applies the bitwise logical operations to audio waveforms, and finds six basic properties of the modulation process. Section 3 validates these properties using a practical XOR based implementation, analyzes the effects of the synthesis parameters to the produced waves, and compares the results with OR and AND operations. Finally, section 4 concludes the paper.

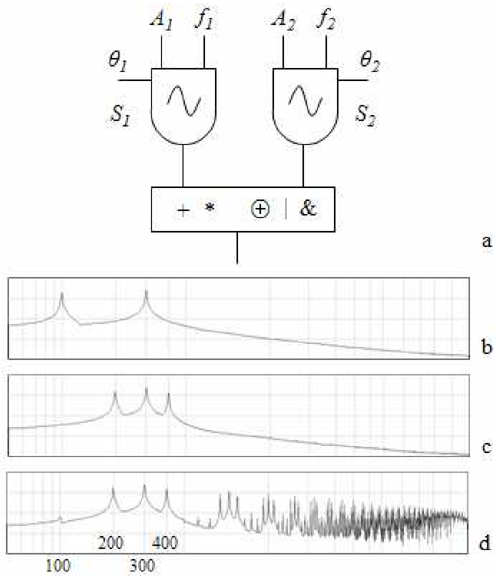

Figure 1: *Elementary operations on sinusoids. a) the instrument (A_1=A_2, f_1=100Hz, f_2=300Hz, θ_1=θ_2=0, S_1 unipolar), b) additive, c) AM, d) XOR (A_2=A_1/100).*

2. BASIC PROPERTIES OF LOGICAL MODULATION

Table 1 shows the familiar binary truth table for basic logical operations OR, AND and XOR [5]. When applied to a vector of bits in a bitwise manner, it is possible to extend the scope of the operations to cover base-2 integer representations, like audio samples.

Table 1: *Truth table for basic logical operations.*

A	B	OR	AND	XOR
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	0

By inspecting table 1, it can be seen that

$$A \text{ OPER } B = B \text{ OPER } A \quad (1)$$

$$A \text{ XOR } B = (A \text{ OR } B) - (A \text{ AND } B) \quad (2)$$

$$A \text{ XOR } B = 0, \text{ when } A = B \quad (3)$$

These properties hold also when operands A and B are replaced by audio waveforms. Equation (2) can be verified also from fig. 2, where sinusoidal source waveforms S_1 and S_2 are drawn with dotted lines, their XOR resultant as a thick line, resultant OR as a thin blue line, and resultant AND using a thin red line. It might be also noted, that the sum of OR and AND resultants equals the sum of S_1 and S_2.

Furthermore, because of (2), and because OR and AND resultants are bounded by maximum and minimum values of source waveforms S_1 and S_2, logical modulation operation can not produce a result value that is larger in magnitude than the maximum value of the source waveforms, i.e.,

$$|\,[S_1 \text{ OPER } S_2](t)\,| \leq |\max[\,S_1, S_2\,]| \quad (4)$$

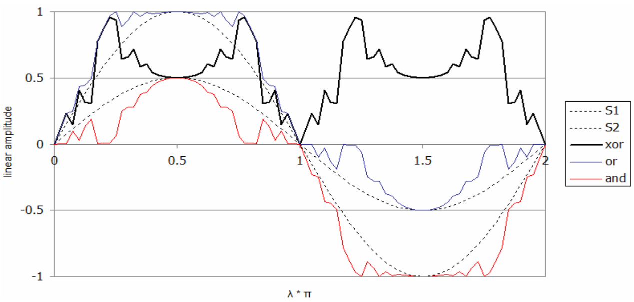

Figure 2: *Logical operations on sinusoidal waveforms. Low sample rate was used to avoid cluttering the figure.*

Fig. 2 also shows that the XORed resultant is full-wave rectified. Although that is a special case (i.e., the waveform is not generally symmetric), it has important implications which shall be discussed in the next section. This property is due to two's complement representation of negative numbers, resulting that XOR operation shares the rule of signs with multiplication operation, i.e., the result is positive if both operands are of same sign. OR operation is positive if both operands are positive, while AND is positive if either operand is positive.

The sixth basic property, also obtainable from fig. 2, is that the resultant waveforms tend to have sharp corners. These are due to local overflows, which occur because bitwise logical operations ignore the carry bit resulting from operations on previous digits.

3. TESTING AND ANALYSIS

In order to validate the basic properties described in section 2, to gain understanding of their implications, and the effects of the synthesis parameters, XOR modulation tests were carried out using the instrument of fig. 1a. There were two identical sinusoidal oscillators S_1 and S_2 (bipolar within a range of -1..1), with independent parameters to control the maximum amplitude A, frequency f, and phase θ of each oscillator.

The synthesized audio files used in the following analysis, and the implementation with source code are available at [1].

3.1. Implementation

Pure Data (PD) [6] patch was devised to realize the instrument of fig. 1a, hosting custom C externals to implement the bitwise logical operations between source signals (the *expr~* external included in PD distribution did not work as expected). Because audio operations in PD are performed in floating point format, and because C language logical operators are only able to handle unsigned integer operands, it was necessary to transform between the two representations. 24 bit samples were used in integer space in order to avoid the problems with overflowed two's complement values[1].

3.2. Implications of the basic logical modulation properties

If all source oscillator parameters are equal, then according to (3), there should be no output sound. The tests revealed that this was indeed the case. The fact should be considered in practical implementations, because the cancellation effect can happen with any set of synthesis parameter states. However, once that was realized in tests, it was relatively easy to tweak the parameters so that there were no dropouts.

The tests validated also that equation (1) was correct. The implication of this is that there is no explicit carrier/modulator relationship in bitwise logical modulation. Equation (2) was not directly evaluated, but it seems that its consequence (4) holds. In other words, it is possible to bound the amplitude level of the synthesized output, so the algorithm is stable.

There were several implications of the sign rule. The rectification creates a sharp corner at half-wave zero crossing, and introduces thus more upper end spectral content. This has some interesting use cases, see for example the rectified sine discussion in section 3.4. It is also possible to remove the full-wave rectification effect of XOR operation by breaking the sign rule inside the logic block of the instrument, and reverse the polarity of the rectified segment. This cancels the octave shift effect of rectification. It should be noted, though, that XOR operation does not necessarily produce symmetric waves.

Although certain synthesis parameter combinations are able to produce band-limited spectra, in general, the sharp edges manifest themselves as aliasing. Because it is possible to detect the positions of the edges, there might be a way to reduce these artefacts. On the other hand, one might argue that because bitwise logical modulation is a digital synthesis technique that is partly characterized by aliasing, it should be treated as an ingredient, and left intact.

3.3. Effects of the amplitude parameter A

Frequencies and phase offsets of both oscillators were fixed (f_1=100Hz, f_2=0Hz, $\theta_1=\theta_2$=0), A_1 was initially fixed to full range (100), and A_2 was varied between zero and full range.

At A_2=0 there was naturally no XOR modulation, and the output consisted of a single sinusoid generated by S_1. When A_2 was increased, a constant offset equal to A_2 was added to the source sinusoid. However, due to the carry-less operation of XOR, the amplitude axis was quantized into A_2/100 ranges. The XOR operation ensures that the superimposed amplitude does not exceed these limits by mirroring the overshoots at each step.

The end result was that the carrier sine wave was decorated by pulse- or ramplike steps (cf. fig. 3a-c). All waves synthesized this way had buzzy pulse-like character with odd harmonics, and

[1] It is possible to optimize the code, because for example the SSE2 processor instruction set has an XORPD instruction, which operates on double floating point numbers directly.

as a special case, by setting A_1=1, the output was a clean square wave (cf. fig 3d). A_1 had the overall effect of straightening out the decorated steps towards more rectangular waveshape. A large number of different waveforms could be produced with different combinations of A_1/A_2.

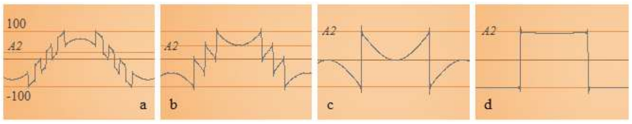

Figure 3: *Constant XOR modulation (f_2=0). a) A_2=25, b) A_2=49, c) A_2=100, d) A_2=100. a-c: A_1=100, d: A_1=1.*

3.4. Effects of the frequency parameter *f*

The parameters of oscillator S_1 were fixed ($A_1=100$, f_1=100Hz, θ_1=0). f_2 was varied between 0..1200 Hz, and the amount of modulation A_2 was first set to low values (0.01..1), and then between 1..100.

As discussed above, by setting $A_2 = 0.01$ and f_2 = 0, there was just one spectral component located at f_1. When f_2 was then increased above zero, the spectrum was spread into lower and upper sidebands, with components located at $f_1 - kf_2$ (k=1,2,..). According to [2, page 76], this is similar to what RM produces for a periodic modulator and a sinusoidal carrier wave. In that case, the RM modulator has to be more complex than a mere sinusoid. With XOR modulation sinusoid is sufficient, and as a result, the generated spectrum is more harmonically related than the complex source RM product.

When f_2 was still increased, the sidebands moved further apart. However, the negative frequencies of the lower sideband were reflected at 0 Hz, causing them to move towards the upper sideband. Eventually, when the upper sideband and the mirrored frequencies overlapped or arranged themselves in a harmonic relation, a steady state tone was produced. These states seemed to occur when the frequency ratio f_2 / f_1 was a rational number (e.g., *n*/5, *n*/4 etc., where *n* is 1,2,...). At higher frequencies, steady states were found at $f_2 = nf_1$.

The fundamental frequency component of the composite sideband spectrum was one of the reflected partials. Usually, the steady state was achieved when the fundamental and the upper part of the spectrum were at odd harmonic relation, producing a pulse-like timbre (the even harmonics were observable, but they had lower amplitudes). However, at $f_2=f_1$ all harmonics were present, and because the modulation amount was very low ($A_2 = 0.01$), the spectrum was almost alias-free, mellow sawtooth produced by a rectified sine waveform (see fig.4 and also [7]).

By increasing the modulation amount A_2 above 1, the behaviour was somewhat different, because even at f_2=0, there was already upper level content in the spectrum (as defined by A_1/A_2, see section 3.3). When f_2 was increased above zero, the principle of two sidebands and mirrored partials was still valid, but the generated spectrum was more complex because of the initial spectral content. Setting f_2<20 Hz (i.e., to control rate), S_2 started operating as a low frequency oscillator, cycling through the XOR generated waveforms of fig. 3 in a stepwise manner. The transition between the waveforms had more of a rhythmic character than a smooth morph, and each cycle was further affected by pops that occurred when the oscillator parameters reached identical states, in accordance to equation (3). At f_2>20 Hz (i.e., at audio rate), there were still steady states at some ratios of f_2/f_1. The most prominent ones seemed to occur at n/2 and n/3 (n=1,2,3..). Fig. 5 shows the spectrum when $f_2=f_1$ and A_2=50, corresponding to the situation depicted in fig.2.

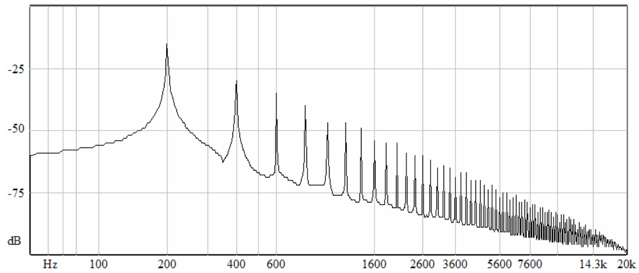

Figure 4: *Rectified sine spectrum.*

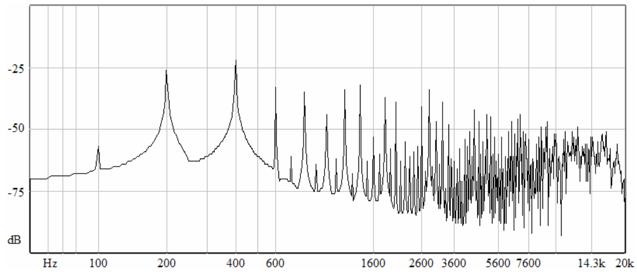

Figure 5: *Spectrum of XOR wave plotted in fig. 2.*

3.5. Effects of the initial phase offset parameter θ

Phase offset θ_1 was fixed to 0, while θ_2 was varied between zero and 2π (corresponding to full wavelength).

Because of the sign rule, and because source waves were sinusoidal, changing the initial phase offset inverted portions of the resultant wave. The symmetry of the source waves affected that between 0..π the inverted portion gradually changed from non-inverted to full polarity flip, and then between π..2π back to the original non-inverted state again.

In between, the effects varied from subtle to dramatic, depending on the values of other parameters. For example, setting phase offset to π/2 in the rectified sine of fig. 4 raised the levels of upper harmonics, making the sound brighter and closer to that of a sawtooth. On the other hand, using the parameters defined in fig. 2, the entire shape of the resultant was affected in more dramatic ways.

3.6. Effects of unipolarity of oscillator S_1

AM employs a unipolar modulator [2]. This will produce three spectral components (f_c–f_m, f_c and f_c+f_m) for a sinusoidal carrier-modulator pair (cf. fig. 1c). If the instrument shown in fig. 1a is modified so that S_1 is unipolar, XOR modulation will produce these three components at the low end of the spectrum, but in addition, there will be more groups of three components at $kf_2 \pm f_1$ (k=1,3,5..), filling the upper side band of the spectrum (cf. fig. 1d). There are also less prominent groups of three at $kf_2 \pm f_1$ (k=2,4,6,..). This is also in accordance with complex source RM. Increasing the modulation amount adds distortion and affects thus also the balance of partials.

3.7. Animation

Phase offset could be easily animated by detuning the oscillators by few cents. Again, the effect was dependant on the settings of other parameters, but detuning amounts below 20 cents produced fat, moving textures particularly on the more distorted resultant waves (like the one shown in figures 2 and 5). The timbres were similar to supersaw/squares of some subtractive synthesizers. Because the texture of these sounds is quite thick, aliasing artefacts were considered tolerable (sound samples are available at [1]). As discussed above, it was also possible to get various rhythmic effects by setting one of the oscillators to operate at lower control rate frequencies.

Modulating the amplitude or frequency of either oscillator S1 or S2 produced similar results. Audio rate AM was able to generate the thickest timbres. Most effective frequency ratios of f_m / f_1 were identical to those found in section 3.4. When audio rate AM was coupled with oscillator detuning, the sound could be considered very rich and evolving (sound samples are available at [1]).

As a final example, fig. 6 shows a smooth morph between pulse – sawtooth – pulse waveforms. The morph was produced by tweaking A_1 between 0-100-0, (A_2=1, f_1=f_2=100 Hz, θ_1= π/2, θ_2=0, S_1 unipolar). The upper part of the spectrum is omitted from the figure for clarity.

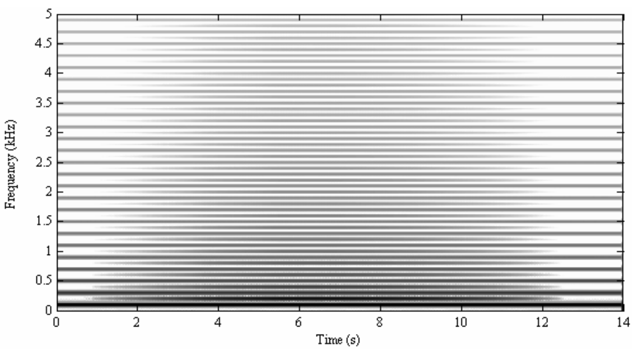

Figure 6: *Morph between pulse – sawtooth - pulse.*

3.8. Comparing XOR with OR and AND operations

Bitwise logical modulation using OR and AND operations share most XOR operation properties and qualities. There are two basic differences, however. First, because of the sign rule differences, OR and AND operations produce only half-wave rectified waveforms. Second, both OR and AND spectrums contain a prominent frequency peak at f_1. That partial is missing from XOR because of equation (2), and is the reason why XOR operation might be considered more interesting from synthesis perspective. Overall, however, OR and AND resultants contain slightly less aliasing artefacts than XOR produced waveforms.

4. CONCLUSION

In this paper we have demonstrated that computationally efficient bitwise logical operations between two sinusoidal source waveforms are capable of producing rich, animated and harmonically related spectra. At low modulation amounts, the methods are able to produce, for example, simple sawtooth and square waves, and to smoothly morph between the two. At higher modulation amounts, a wide variety of more complex waveforms can be generated, including those resembling fat supersaw/square timbres that would otherwise require larger number of interacting oscillators.

The spectrum consists of two layers. In the first layer the components are positioned as in ring modulation (XOR) or as in amplitude modulation (OR, AND) with complex modulator and sinusoidal carrier. The second layer is produced by distorted waveforms, and it is located at the upper end of the spectrum. The second layer is not present when modulation amount is very small, but for higher values, it produces aliasing. However, aliasing is considerably smaller with sinusoidal source waves than when more complex input sources were to be used.

We estimate that the most obvious application of bitwise logical modulation synthesis is in the production of rich source material for further processing. The benefits are in the efficient production of both simple and complex harmonic waveforms. The disadvantages include lack of generality and aliasing.

Further work is needed on determining the structure of the upper end of the distortion spectrum and its relation to synthesis parameters. Also, interesting work could be done by studying more complex logical expressions and their effect on the produced spectrum.

5. ACKNOWLEDGMENTS

The author would like to thank Prof. Vesa Välimäki for helpful comments on the paper. This work has been co-funded by the European Union (EU) as part of the 6th Framework Research Programme with the project CALLAS (ref. 034800).

6. REFERENCES

[1] Sound samples and source code http://www.hiit.fi/u/kleimola/dsp/blm/logmod.html

[2] U. Zölzer, Ed., *DAFX – Digital Audio Effects*, J. Wiley & Sons, 2002.

[3] J. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation", *J. Audio Eng. Soc.*, vol. 21, No. 7, pp. 526-534, September 1973.

[4] Arp Instruments, *ARP Odyssey Service Manual*, Available http://www.hylander.com/moogschematics.html, Accessed June 08, 2008.

[5] J. Millman, *Microelectronics*, McGraw-Hill, 1979.

[6] *Pure Data portal*, http://puredata.info, Accessed June 08, 2008.

[7] J. Lane, D. Hoory, E. Martinez, and P. Wang, "Modeling analog synthesis with DSPs", *Comput. Music J.*, vol. 21, no. 4, pp. 23–41, 1997.