A NEW SHADE OF PINK

Stefan Stenzel*
Waldorf Music GmbH, 53424 Remagen, Germany
(Dated: August 8, 2014)

A new method for efficient and accurate generation of $1/f$ noise, or pink noise, is shown. This new method performs well in respect to signal quality while requiring surprisingly few calculations for a multirate algorithm.

INTRODUCTION

Pink noise, or $1/f$ noise, has a frequency rolloff of 3 dB per octave, which for many audio purposes is often the preferred choice because it is perceived as wide band noise. A conventional way for generating pink noise digitally is to filter white noise, but unfortunately a 3 dB per octave filter design is not straightforward, so it comes with a compromise between performance and accuracy.

Another approach, commonly referenced as Voss-McCartney algorithm [1], uses a sum of several white noise signals with octave spaced sample rates, upsampled by zero order hold interpolation. This produces considerable aliasing, but as the goal is to produce noise this can be tolerated. The spectrum still emits some considerable deviation from the ideal $1/f$ slope, as shown in [2] and Fig.4.

If linear interpolation, also known as first order hold, is used for upsampling, most of the spectrum comes much closer to the desired 3 dB rolloff. Only the higher end of the spectrum requires special attention, which can be solved by applying a simple filter to an additional white noise signal. All of this might seem rather complex compared to prevailing implementations, but with the proper choice of white noise, all condenses to an algorithm with very few instructions and without any multiplication, division or transcendental functions, which makes it also a good candidate for VHDL implementation.

WHITE NOISE

A linear feedback register, commonly abbreviated LFSR, emits a maximum length sequence, MLS, of pseudorandom bits if the feedback terms are chosen properly. This sequence of bits is spectrally flat, which makes it an excellent choice for a white noise signal. As the bits are uncorrelated, it can also serve as a source of random bits for a plurality of white noise signals, a property we will exploit subsequently. The sequence repeats after $2n-1$ cycles, with n representing the number of bits of the LFSR. With a sample rate of 44.1 kHz, a 32-bit LFSR runs about one day before repeating, which should be good enough for most audio related purposes. The choice of a single bit wide noise signals allows for multiple simplifications in the presented algorithm.

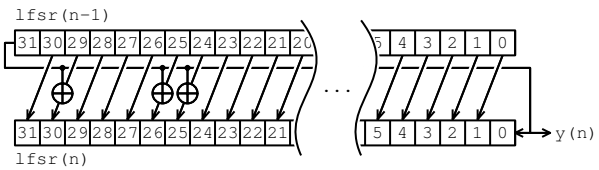

FIG. 1: LFSR used as source for pseudorandom bit sequence

UPSAMPLING

Starting with K pseudorandom bit sequences s_k with bits interpreted as either -1 or 1, K new upsampled sequences u_k at our output sample rate are generated by inserting $k-1$ zeros after each bit of s_k. With these zeros in mind, it is

easy to see that the power of such signal follows

$$P_{u_k} \equiv 2^{-k} \tag{1}$$

The zero order hold sequences g_k are given by

$$g_k(n) = \sum_0^n u_k(n) - u_k(n - 2^k) \tag{2}$$

A similar second step gives the linear interpolated sequences y_k, the only difference being the correction factor 2^{-k} as compensation for the growth from integration:

$$y_k(n) = 2^{-k} \sum_0^n g_k(n) - g_k(n - 2^k) \tag{3}$$

With the complete transfer function given by

$$Y(z) = 2^{-k} \left[\frac{(1 - z^{-2^k})}{(1 - z^{-1})} \right]^2 \tag{4}$$

It is no coincidence that this looks almost identical to a 2nd order CIC upsampler where the gain correction 2^{-k} is often omitted or applied in a subsequent stage in CIC formulations. Because we perform upsampling from multiple rates, it is essential to keep it right here.

Finally the squared magnitude for the sum of all upsampled noise sources can be shown to be

$$|H(z)|^2 = P_{u_k} |Y(z)|^2 \tag{5}$$

Which leads to

$$|H(z)|^2 = (1 - z^{-1})^{-4} \sum_{k=1}^{K} 2^{-3k} (1 - z^{-2^k})^4 \tag{6}$$

As can be seen in Fig. 2, both low frequency and higher frequency ends differ significantly from the ideal $1/f$. A lower threshold is required, otherwise the resulting signal could not be represented digitally as $\lim_{f \to 0} 1/f = \infty$. For audio, frequencies below 10 Hz are usually not audible, so we set this as our lower limit. The high frequency end however is not yet accurate enough, neither with y_k starting from $k = 0$ nor from $k = 1$. For refining the high end, we employ

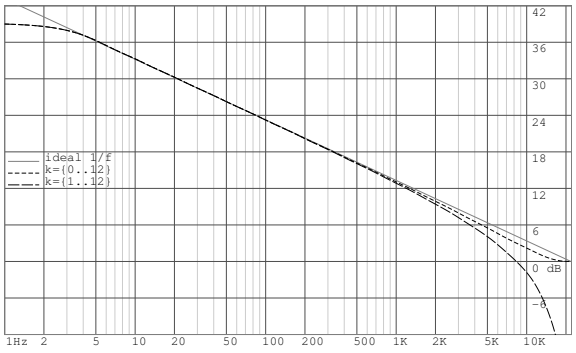

FIG. 2: The diagram shows $|H(z)|$ in dB for sums of upsampled pseudorandom noise signals y_k for $k = \{0..12\}$ and $k = \{1..12\}$ compared to the ideal 1/f curve. Sampling rate fs is 44.1 kHz, spectrum goes from 1 Hz to $fs/2$ =22050 Hz.

a finite impulse response filter on an additional pseudorandom noise signal which approximates

$$|F(z)|^2 \approx |H(z)|^2 - \frac{1}{(1 - z^{-1})} \tag{7}$$

Using standard FIR design methods, coefficients for a suitable 12-tap FIR filter have been evaluated. With this filter, the relative error in the band of interest from 10 Hz to 22050 Hz is 0.04 dB.

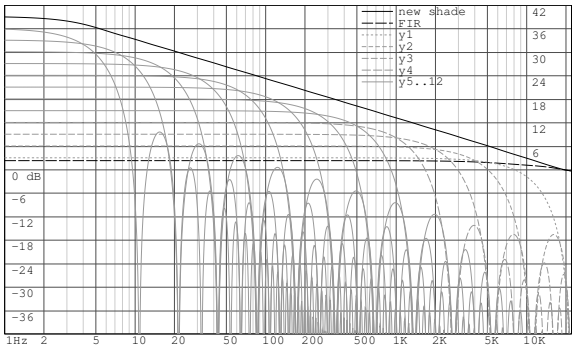

FIG. 3: spectra of complete pink noise approximation (solid line) as sum of 12-tap FIR filtered noise (dashed) and spectra of individual upsampled noise signals (gray/dashed). The sampling rate fs is 44.1 kHz, shown spectrum goes from 1 Hz to $fs/2$ =22050 Hz.

IMPLEMENTATION

So far it has been demonstrated how to sum up some generated signals to get an acceptable approximation to pink noise. Here I will introduce a way to calculate this quickly with surprisingly few operations. If Eq.3 is rewritten as

$$y_k(n) = \sum_{0}^{n} 2^{-k} g_k(n) - 2^{-k} g_k(n - 2^k) \quad (8)$$

we keep in mind that the input sequence g_k is a single bit signal, $2^{-k} g_k$ can be represented by a fixed point number with only one signigficant bit at position $K-k$. Both terms of the summation can be combined into a fixed point number, called *inc* and *dec*:

$$inc(n) = \sum_{1}^{K} 2^{-k} g_k(n) \quad (9)$$

$$dec(n) = \sum_{1}^{K} 2^{-k} g_k(n - 2^k) \quad (10)$$

The terms *inc* and *dec* conveniently contain all K first order hold upsampled noise signals g_k with just one bit for each k. Because k is a negative exponent, bit k is located at position $K-k$, making both *inc* and *dec* bit-reversed compared to the standard binary number representation.

As the state for each g_k changes every 2^k samples, we can use a scheme where only one bit position is updated for each output sample as long as $k \neq 0$. For binary operations, a bitmask is used where bit k is set for each iteration as shown in Table I.

For implementing the filtered portion that is subsequently added, we first precalculate all possible outputs of the FIR filter. For a 1-bit input and a filter length of M, there are 2^M possible filter outputs. This becomes clear if we look at the input sequence as a binary number of length M. The output is generated by using the last M pseudorandom bits as an index into the table of recalculated filter outputs. With our choice of LFSR from Fig.1, the lower 26 bits contain the undisturbed sequence of the last generated bits. As long as the filter length does not exceed 26, we can exploit this property and take a portion of these bits as index for the table of precalculated FIR values $firtab$. This is sufficient for our choice of a filter length of 12 taps.

TABLE I: Sequence of k and bitmask to update *inc* and *dec*

n	k	bitmask
1	1	0x800
2	2	0x400
3	1	0x800
4	3	0x200
5	1	0x800
6	2	0x400
7	1	0x800
8	4	0x100
9	1	0x800
10	2	0x400
...		
4095	1	0x800
4096	12	0x001
4097	1	0x800

So for each sample, the operations to generate one pink noise sample are:

1. move bit k of *inc* to bit k of *dec*
2. update LFSR and get random bit r
3. move random bit r to bit k of *inc*
4. perform $sum = sum + inc - dec$
5. $output = sum + firtab$[LFSR & 0x0FFF]

RESULT

As shown in the previous section, our new method for generating $1/f$ noise is not very complex. In comparison, algorithms applying recursive filters to white noise require significantly more operations than out method.

TABLE II: Comparison of our algorithm and several other methods for generation of $1/f$ noise found in [3], except [simple] which is measured from [4], the self-proclaimed "best free color noise generator on the Internet". Estimation of relative complexity is based on comparison of operation count.

method	error dB	complexity
voss	1.83	8
rbj	0.61	3
pk3	0.06	30
pke	0.91	15
simple	2.4	?
new shade	0.04	1

The original Voss-McCartney algorithm [1] exhibits a large error at a relatively high computational cost.

An implementation in C++ of the proposed algorithm under a permissive license is available from [5].

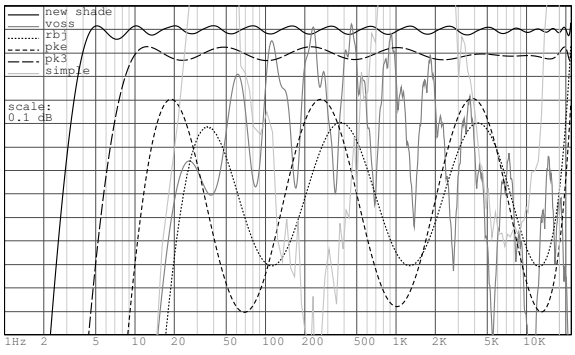

FIG. 4: relative error in dB of several methods. The absolute y-axis position of each method is chosen arbitrarily for better visualisation.

DISCUSSION

A new algorithm for generating $1/f$ noise has been established. Unfortunately it is not error-free, but it exhibits a good ratio of error vs. complexity.

A scheme for 2nd order multirate CIC upsampling of 1-bit signals has been established. This scheme can be extended to higher orders and signals with more bits, proof of this is beyond the scope of this paper.

Additional research could improve the signal quality, a higher order upsampling or "sharpened" CIC filters could very well lead to less error.

Extending the bandwidth towards low frequencies is easy by increasing K, the number of noise sources. This extension comes free in regard to operation count, but it requires more bits and more headroom, the accumulator *sum* needs to be $K\lceil log2(K)\rceil$ bits wide to avoid overflow.

Given the relative ratio of error vs. complexity, this new algorithm should be prefered over prevailing methods.

* stefan.stenzel@waldorfmusic.de

[1] M. Gardner, *"White and Brown Music, Fractal Curves and One-Over-f Fluctuations"*, Scientific American, 288, 16 (1978)

[2] Sophocles J. Orfanidis, *"Introduction to Signal Processing"*, Rutgers University, p. 733 (1996)

[3] http://www.firstpr.com.au/dsp/pink-noise

[4] http://simplynoise.com

[5] https://github.com/Stenzel/newshadeofpink