Detection and Removal of Limit Cycles in Sigma Delta Modulators

Joshua D. Reiss and Mark Sandler, *Member, IEEE*

***Abstract*—Sigma delta modulation is a popular method for converting signals from analog to digital and vice-versa. However, sigma delta modulators (SDMs) may suffer from limit cycles, where the output bits may enter a repeating pattern. Current methods of preventing this phenomenon introduce unwanted noise, do not always succeed, and are often implemented when not needed. We present a more effective method for detecting and removing unwanted limit cycles. This method includes adding a small disturbance to the input, which destroys the periodicity of sigma-delta analog-to-digital conversion (ADC) modulator's output sequence and thereby removes the limit cycles. Compared with conventional methods this method is simpler to implement, and the SDM has less signal-to-noise ratio (SNR) penalty and a higher allowed input dynamic range. Various implementations of the limit cycle detection and removal schemes are described for feedforward SDMs. Results are reported which demonstrate the success of these methods.**

***Index Terms*—Analog-to-digital conversion (ADC), digital-to-analog conversion (DAC), limit cycles, sigma delta modulation.**

I. Introduction

SIGMA delta modulation is a popular method of converting signals from analog to digital and vice-versa. It typically involves converting a signal into a low-bit, highly oversampled representation. It benefits greatly from the oversampling in that a feedback path may be used to shape the quantization noise into high frequencies where it is not noticeable. Due to its low circuit complexity and robustness against circuit imperfections, low bit sigma delta-based analog-to-digital and digital-to-analog converters [(ADC) and (DAC)] are widely used in audio applications, such as cellular phone technology and high-end stereo systems. However, sigma delta modulators (SDMs) suffer from limit cycles, where the output bitstream may enter a repeating pattern with frequency components that were not present in the input signal.

Though limit cycle operation may be exploited to reduce power consumption and design complexity [1], limit cycles are generally unwanted since they result in frequency components in the output bitstream which were not present in the input signal [2]. Digital filters may be designed so as to minimize the occurrence of limit cycles [3], but it is not yet clear if this approach may be extended to limit cycles in SDMs. In practice, limit cycle prevention is typically achieved by adding a random signal, with a uniform, triangular or spectrally shaped probability distribution, just prior to quantization [4], [5]. When this "dithering" sequence is added, an output bit may be flipped (output bit changed from $+1$ to -1, or from -1 to $+1$), and the periodic output pattern might be destroyed. However, the dither decreases the signal-to-noise ratio (SNR), the stability, and the dynamic range of the SDM. Furthermore, it is often added when it is not needed, and in many situations may not be sufficient to destroy a limit cycle. Dither is more effective in the prevention of noise modulation, where its application has a sound basis in theory [6]. Thus, the aim of this paper is to exploit theoretical understanding of limit cycles in order to devise a better method for their prevention.

Fundamental work on limit cycles in SDMs has usually been constrained to low-order SDMs [7]–[9], and, hence, is of little practical value to engineers who use high-order noise shaping techniques. Recent work has significantly advanced the theory of limit cycles in SDMs [2], [10]–[13]. Most notably, in [2], results were derived concerning the character of limit cycles for a general feedforward (also called interpolative) SDM, and on their stability in particular. In [13], similar results were obtained for feedback SDMs.

This understanding has been exploited to devise effective methods for detecting and removing limit cycles [14]. Here, we will describe full details of their implementation, provide analytical justification for their effectiveness, and characterize their behavior through theory and C source code simulations. The paper is organized as follows. In Section II, the mathematical framework, based on a state space description of the SDM, is presented. In Section III, methods are presented to detect limit cycles in feedforward SDMs. Properties of the limit cycle removal methods are also described in this section, such as the time to detect a limit cycle and the probability of false detection. In Section IV, the limit cycle removal method is described, and a proof is provided which shows that this method is guaranteed to remove limit cycles in traditional feedforward SDMs. Section V discusses implementation details, including preferred parameter settings and techniques for circuit implementation. Finally, the conclusion summarizes the results, highlights the key features of the technique and discusses how these methods may be adapted to other types of SDM design.

II. State Space Description

For the analysis that follows, we will restrict ourselves to discussion of feedforward SDMs without resonator sections. The limit cycle detection and removal mechanisms to be described in later sections may also be valid for feedback designs, resonator sections, and other modifications, but analysis of this design is particularly tractable.

Manuscript received October 26, 2007; revised February 17, 2008. First published April 18, 2008; current version published November 21, 2008. This paper was recommended by Associate Editor T. B. Tarim.

The authors are with the Centre for Digital Music, Queen Mary, University of London, London, E1 4NS, U.K. (e-mail: josh.reiss@elec.qmul.ac.uk; mark.sandler@elec.qmul.ac.uk).

Digital Object Identifier 10.1109/TCSI.2008.925078

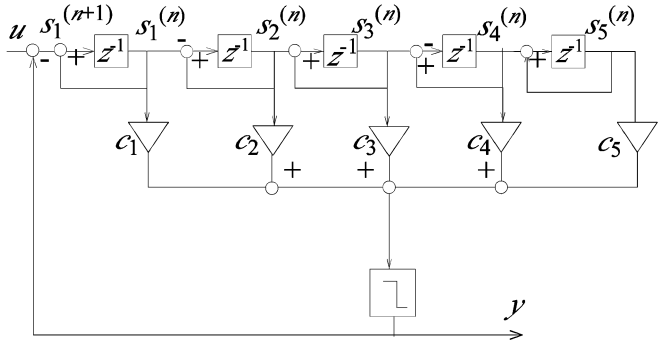

Fig. 1. States in a fifth-order SDM.

A convenient way to describe the time domain behavior of an SDM is the state space description. This represents the state of the SDM at any time as a matrix operation applied to the state at the previous clock cycle. The power of the state space description is that it allows us to create a very compact description of the state of the SDM from time $t = 0$ to time $t = n$.

For an N^{th}-order feedforward (or iterative) SDM

$$\mathbf{s}(n+1) = \mathbf{A}\mathbf{s}(n) + (u(n) - y(n))\,\mathbf{d} \tag{1}$$

where u is the input at iterate n, and y is the output, $+1$ or -1, determined by

$$\begin{aligned} v(n) &= \sum_{i=1}^{N} c_i s_i(n) \\ y(n) &= sgn(v(n)). \end{aligned} \tag{2}$$

This description gives the state of the SDM in terms of a transition matrix $\mathbf{A}$ applied to the previous state vector, and a vector $\mathbf{d} = [10\ldots0]^T$ applied to the scalar quantization error $u(n) - y(n)$.

Fig. 1 gives an example of a typical fifth-order SDM. The coefficients $\mathbf{c}$ determine the noise shaping characteristics, and the loop around each delay z^{-1} represents an integrator.

For this fifth-order modulator, $\mathbf{d} = [1, 0, 0, 0, 0]^T$ and the transition matrix is

$$\mathbf{A} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \tag{3}$$

The compact representation gives the means to directly view the consequences of a limit cycle. If the limit cycle has period P we have, by definition

$$y(n+P) = y(n). \tag{4}$$

An important assumption in earlier work [15] is that periodic behavior in the output of the quantizer implies periodic behavior in the state space variables. In [2], it was proven that, in general, a limit cycle in the output bitstream exists if and only if there is a limit cycle in the state space variables. That is, (4) is equivalent to

$$\mathbf{s}(n+P) = \mathbf{s}(n). \tag{5}$$

Although in its pure definition, a limit cycle is a periodic pattern of infinite duration, in practical situations finite duration periodic sequences, as characterized in [10], can also be problematic. They cause unwanted peaks in the short time power spectrum, and if persistent for significant duration, may cause audible clicks when SDMs are used in audio applications [16]. Thus, a limit cycle detection and removal algorithm should be successful even if (4) is only true for fixed duration, i.e., we must be able to detect and remove limit cycles when (5) is approximately true for a finite time.

III. Limit Cycle Detection

A. State Space-Based Limit Cycle Detection

Equation (5) provides a simple method of determining if a limit cycle exists. At a given iterate which we set to 0, $\mathbf{s}(0)$ may be stored in a buffer. For each successive iterate, $1, 2, \ldots i, \ldots$, up to some value, the buffer duration $P_{\max}, \mathbf{s}(i)$ is computed. If constant input is applied and, for some $i, \mathbf{s}(i) = \mathbf{s}(0)$, then the theorem described earlier guarantees that a limit cycle of period P exists.

This method, while exact, has three drawbacks. First, it requires that a vector of size N be stored. At each time iterate, up to N comparisons must be made. This is unnecessarily complicated. Second, and more importantly, it does not allow for a simple method of making approximate comparisons. When the state space variables are very close to a limit cycle condition, periodic output may be sustained long enough to be problematic. An appropriate measure of the required proximity of the state space variables needed for temporary limit cycle behavior is not obvious and may not be simple to compute.

To alleviate these difficulties, we propose computing a single scalar quantity at each iterate. Note that the effect of a small change in the state space variables tends, over time, to yield a larger change in the later state space variables. The cumulative nature of the integrators implies that s_N varies far more rapidly than any other state space variable.

From (1), if a small perturbation $\boldsymbol{\delta}(0)$ is applied to the state space variables at time 0, then that perturbation grows at a rate given by $\boldsymbol{\delta}(n) = \mathbf{A}^n\boldsymbol{\delta}(0)$. Repeated application of the transition matrix $\mathbf{A}$ yields the binomial coefficients [2]

$$\mathbf{A}^n_{ij} = \begin{cases} 0, & \text{if} \quad i < j \\ \dbinom{n}{n-(i-j)}, & \text{otherwise.} \end{cases} \tag{6}$$

Thus, the terms of $\boldsymbol{\delta}(n) = \mathbf{A}^n\boldsymbol{\delta}(0)$ are given by

$$\begin{aligned} \delta_i(n) = \binom{n}{n}\delta_i(0) + \binom{n}{n-1}\delta_{i-1}(n) \\ + \cdots + \binom{n}{n-i+1}\delta_1(n). \end{aligned} \tag{7}$$

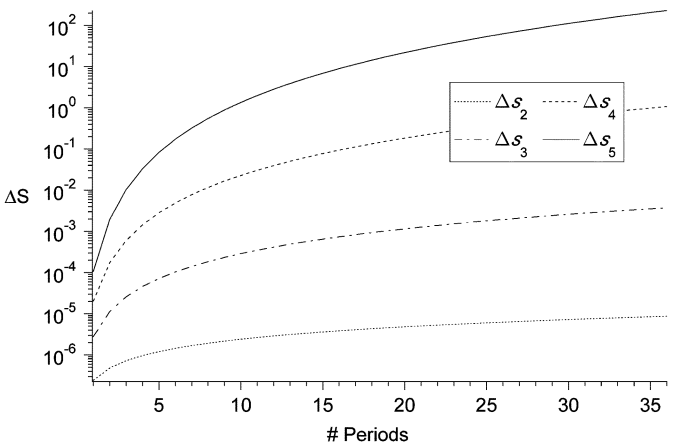

Fig. 2. For a fifth-order SDM, given by (3), this depicts the growth of a disturbance to the state space variables as a function of the number of periods.

For lowpass SDMs, typically $\delta_1(n) < \delta_2(n) \ldots < \delta_N(n)$. This implies that when the system is near a limit cycle, there may be considerable variability in s_N compared with $s_1, s_2, \ldots s_{N-1}$. As an example, consider the fifth-order SDM given by [2]

$$\begin{aligned} c_1 &= 0.5761069262, c_2 = 0.1624753515, c_3 \\ &= 0.0276093301, c_4 = 0.0028053934, c_5 \\ &= 0.0001360361. \end{aligned} \tag{8}$$

For a sampling rate of 64×44.1 kHz, which is often used in audio applications [17], this has a corner frequency of 80 kHz.

Fig. 2 depicts the growth of each variable for a small perturbation (10^{-8}) away from a period 24 limit cycle. One can clearly see the higher growth rate for the later state variables.

Note that this behavior is qualitatively independent of the choice of the coefficients $c_1, c_2, \ldots c_N$. The coefficients typically differ greatly in value. For a lowpass SDM, $c_1 > c_2 > \cdots > c_N$. This also implies that s_N may vary greatly with the output bitstream unaffected.

Thus, when the system is near a limit cycle, $s_1, s_2, \ldots s_{N-1}$ will vary almost periodically whereas s_N will diverge away from periodic behavior.

Furthermore, if we define $\mathbf{s}^*(0) = \mathbf{s}(0) + (0 \cdots 0\delta)^T$ then we have

$$\begin{aligned} \mathbf{s}^*(1) &= \mathbf{A}\mathbf{s}^*(0) + (u(0) - y(0))\mathbf{d} \\ &= \mathbf{s}(1) + (0 \ldots 0\, \delta)^T. \end{aligned} \tag{9}$$

That is, a perturbation applied only to the last state variable does not grow and does not affect the other state variables.

s_N is, thus, ignored in computing whether we are near a limit cycle. By not comparing s_N, we can find short term limit cycles where the state space variables are not exactly repeating but the output bitstream may remain periodic long enough to be problematic.

In order to have a scalar quantity for comparison, as well as to take into account the differing sizes of the variables, we use as our stored variable

$$f(n) = \sum_{i=1}^{N-1} c_i s_i(n) = v(n) - c_N s_N(n). \tag{10}$$

Thus, if a limit cycle of period P exists

$$f(n+P) = f(n). \tag{11}$$

In order to account for approximate limit cycles, we apply a tolerance, T. For an initial iterate, we store the value $f(0)$ and check each successive iterate to see whether (11) holds. Thus, our condition for a limit cycle of period P to be observed is

$$|f(P) - f(0)| < T. \tag{12}$$

If it does, then we apply the limit cycle removal algorithm to be described in Section IV. The tolerance only serves to detect approximate limit cycles, but does not affect whether an ideal limit cycle is detected.

After a number of iterates $P_{\max}$, we reset the buffer from $f(0)$ to $f(P_{\max})$. This allows one to identify limit cycles which appear at later iterates. However, it also implies that we may not identify some limit cycles of period $P > P_{\max}$. The value of $P_{\max}$ is set quite high so as to identify long period limit cycles with a fundamental frequency within the passband, but a maximal value should be only a small multiple of the oversampling ratio for two reasons. First, it has been observed that the occurrence of long period limit cycles is exponentially rare relative to the number of allowable period bit sequences of that same length.[2]. Thus, limit cycles of a frequency much less than the passband cut-off frequency can be safely ignored. Second, if $P_{\max}$ OSR then a limit cycle with period $P < P_{\max}$ may sometimes persist for long enough to be problematic. Evidence of the weak dependence of limit cycle detection on the choice of this parameter setting is provided in Sections III-C–E, and unless stated otherwise, $P_{\max} = 5\text{xOSR}$ was used in the results provided.

This limit cycle detection procedure may be continued indefinitely. The method is robust to the choice of parameters and may be used to detect any limit cycle. A block diagram depicting this method is given in Fig. 3.

Returning to the fifth-order SDM given by (8), with an input of 0.7 and initial conditions $\mathbf{s} = 0$, it exhibits limit cycle behavior. Fig. 4 depicts a time series of the input to the quantizer.

Fig. 4 may be contrasted with Fig. 5, which depicts the function f, as defined in (10), as a function of the iterate. One can see that, when approximate limit cycle behavior occurs, although the quantizer input may drift (as in Fig. 4), $f(n)$ remains periodic.

In Fig. 5, the circled points represent those where limit cycle behavior has been identified, according to (12) and the procedure described earlier. The time lag between the start of the limit

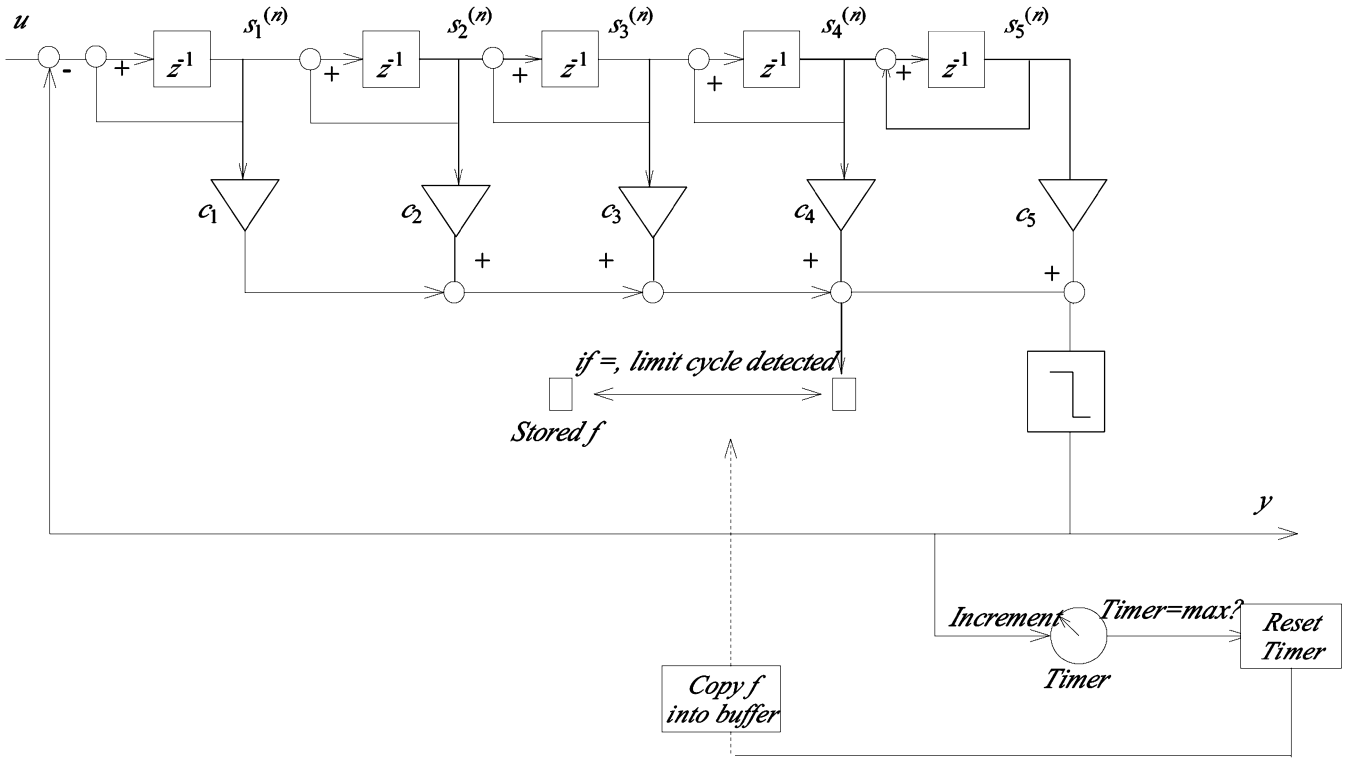

Fig. 3. Block diagram of a fifth-order SDM with a bitstream-based limit cycle detector.

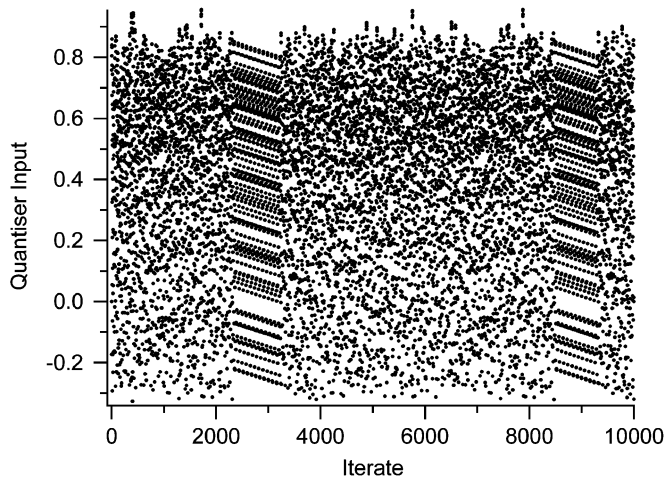

Fig. 4. Plot of the input to the quantizer as a function of the iterate.

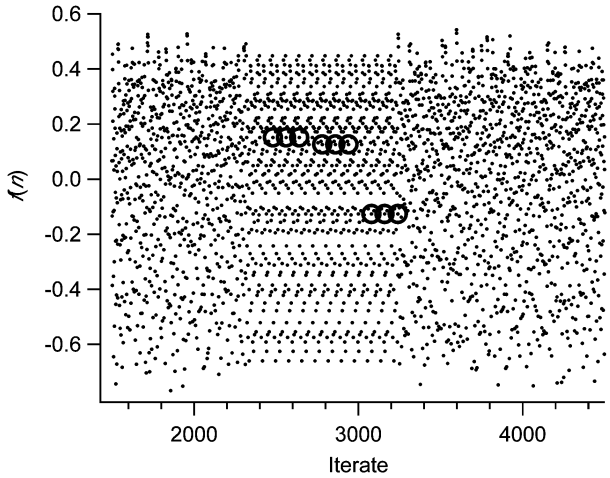

Fig. 5. Plot of f as a function of the iterate. The circled points represent where limit cycle behavior has been identified.

cycle and the identification of limit cycle behavior is due to the choice of parameters in the detection method and due to the fact that a limit cycle may be defined by its repetitive nature, which is not observed until after several repetitions. In this particular case, the limit cycle has been running for approximately 200 iterates before the buffer is reset to a value for f which represents the limit cycle.

B. Bitstream-Based Limit Cycle Detection

Shift registers and bit comparisons are often much easier to implement than any circuitry (analog or digital) for comparison of real numbers or voltages. Furthermore, one can conceive of situations where it is easier to access the output bitstream than the state space variables. Thus, we must consider methods of detecting limit cycles using bit comparisons alone.

A naïve approach would be to implement many shift registers, each one representing periodic output for a different limit cycle. The current output could then be compared with each shift register to see if it appears that limit cycle behavior is occurring. However, to identify all limit cycles of period P this would require on the order of 2^P shift registers, and 2^P comparisons [12].

Thus, an alternate approach is used which minimizes both circuit complexity and the required number of operations. Since a limit cycle represents a repeating pattern in the bitstream, it

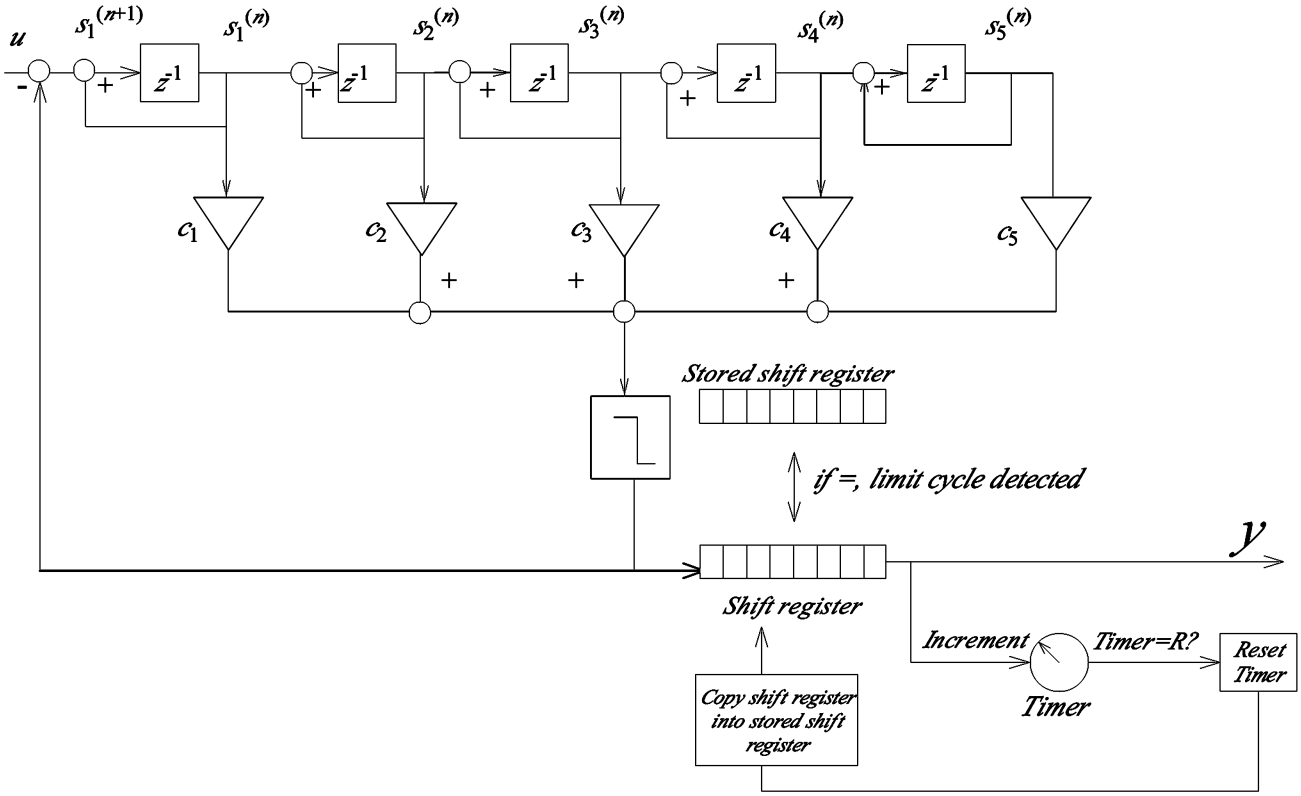

Fig. 6. Block diagram of a fifth-order SDM with a bitstream-based limit cycle detector.

suffices to identify bitstream repetition. To do this, we compare two shift registers; one of the current bitstream output and one of the bitstream output at a previous iterate. For this method, there are two parameters, a shift register length $\mathrm{SR}_{\mathrm{length}}$ and a shift register duration or persistence (the number of iterates until the stored shift register is reset) $P_{\max}$.

The algorithm is as follows:

- Keep a shift register $\mathrm{SR}_{\mathrm{current}}$ of the last $\mathrm{SR}_{\mathrm{length}}$ output bits.
- After every $P_{\max}$ iterates, copy the shift register SR into the shift register $\mathrm{SR}_{\mathrm{stored}}$.
- If, at any iterate, the samples stored in the two shift registers exactly match, $\mathrm{SR}_{\mathrm{current}} = \mathrm{SR}_{\mathrm{stored}}$, then a limit cycle has been detected.

A block diagram of the limit cycle detection system, including the noise shaping, is given in Fig. 6. Note that comparison of bits in the shift register, as well as copying of bits between shift registers, can be done in parallel. Thus, there is no need for any operation to be performed faster than the sampling frequency of the SDM.

C. Time to Detect a Limit Cycle

Of concern with both state space-based and bitstream-based detection is the time it takes to detect a limit cycle from when it first appears, and how frequently it is detected when we are operating in a limit cycle. The time to detect the limit cycle provides a minimum amount of samples that the limit cycle is in existence, regardless of the removal mechanism, and how frequently the limit cycle is detected provides information regarding when and how often the removal method will be applied. We will consider both time to detect and frequency of detection for both techniques.

For state space-based detection, there are two parameters, the buffer duration $P_{\max}$ and the tolerance. If we are in a limit cycle defined by (4) then (5) guarantees that the limit cycle is detected for any value of the tolerance if the limit cycle has period $P \leq P_{\max}$. Lets suppose we are in a limit cycle of length P, where $P_{\max} < P$. At some iterate n the stored buffer is set. This value will, thus, be in the current buffer next at time $n+P$, but before that can be identified, the buffer is reset at iterate $n + P_{\max}$. Thus, limit cycles of period greater than the buffer duration will not be detected.

For $P \leq P_{\max}$, exactly when the limit cycle is first detected is determined by when the buffer is first reset. If the limit cycle begins at iterate 1, then the buffer may be reset at any iterate between 1 and $P_{\max}$. The limit cycle is then detected exactly P iterates later. So the time to detect a limit cycle of period P, from the moment it first occurs, is within the range

$$P + 1 \leq t \leq P + P_{\max}. \tag{13}$$

For, the bitstream-based detection method, the shift register is storing output entirely from the limit cycle only after $\mathrm{SR}_{\mathrm{length}}$ iterates. Limit cycles of a period longer than $\mathrm{SR}_{\mathrm{length}}$ are still identifiable, since this also requires matching of the recent bitstream with the stored values in the buffer. For a limit cycle which begins at iterate 1, the current shift register is first in the limit cycle at iterate $\mathrm{SR}_{\mathrm{length}}$. Ignoring false detections, the time

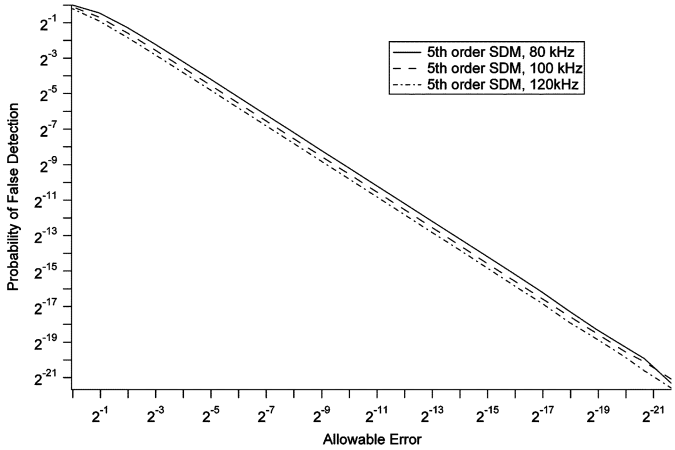

Fig. 7. The probability of falsely detecting a limit cycle as a function of the allowable error for state space-based detection.

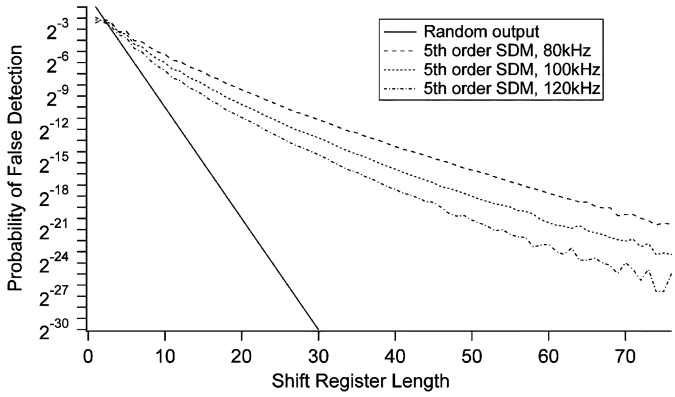

Fig. 8. Probability of falsely detecting a limit cycle as a function of shift register length. In each case, the shift register duration was set equal to the shift register length.

to first detect is thus given as with the state space-based technique, but with an offset of $\mathrm{SR}_{\mathrm{length}} - 1$ iterates. Thus, the time required to detect the limit cycle becomes

$$P + SR_{\mathrm{length}} \leq t \leq P + SR_{\mathrm{length}} - 1 + P_{\max}. \qquad (14)$$

D. Frequency of Limit Cycle Detection

The frequency of limit cycle detection may be found in exactly the same manner for both detection mechanisms. As we shall see, it is dependent on only the period of the limit cycle which is being detected, and the duration $P_{\max}$ of the stored buffer or shift register. Thus, the following discussion is equally applicable to both situations and for the sake of simplicity we will refer to the buffer and buffer duration.

Assume that we are in a limit cycle of length $P \leq P_{\max}$ and a limit cycle has been detected at iterate 0. If the buffer is not reset over the next P iterates, then the conditions for limit cycle detection are next satisfied at iterate P. If on the other hand, the buffer is reset at some iterate R to the current value from iterate $R-1$, then $R \leq P \leq P_{\max}$. So a limit cycle will next be detected at iterate $R - 1 + P < 2P$. Therefore, a limit cycle is detected at most once per period, and at least once every other buffer duration $P_{\max}$. In fact, it can be shown that if P is a divisor of $P_{\max}$, then the limit cycle is detected exactly once every period. It is detected at least once per duration if the period is less than the duration but not a divisor of the duration.

E. Probability of False Detection

1) False State-Space Based Detection: It is possible that (11) may hold when limit cycle behavior is not happening, but would be very rare. It occurs with probability on the order of the digital precision of the hardware, e.g., if there are 2^{16} bits precision and f ranges over values from $-A$ to A, then it would occur with probability close to $1/(2^{17}A)$.

However, the tolerance T in (12) provides a reasonable measure of how often false detections may occur in practice. In Fig. 7, the probability of false detection is depicted as a function of the tolerance. Each data point was generated using 100 output bitstream sequences, each 1 million points long (after initial startup transients were removed), and where each sequence has as input a sinusoid with a randomly generated frequency between 80 and 130 kHz, and random initial conditions. It can be clearly seen that the probability of a false detection has a linear dependence on the threshold T. This implies that f, as given by (10), is approximately uniformly distributed over its full range of allowable values. More aggressive noise shapers typically have larger coefficients, and, thus, have a larger range of internal integrator states. This accounts for the slightly lower probability of false detection for the more aggressive noise shaping.

2) False Bitstream-Based Detection: For bitstream-based SDM detection, false detections are very rare since this requires an exact match over a large number of bits. If the output is truly random, then false detections occur with a probability $2^{-\mathrm{SRlength}}$. However, the output is far from random. This is partly due to the fact that the input is not random (bandlimited, with amplitude safely within stability limits), but also because the sigma delta modulation prevents certain output bit sequences from occurring, regardless of input [12].

Fig. 8 depicts false detections of limit cycle behavior as a function of the shift register length, where shift register duration was set equal to shift register length. Data were generated using the same signals as used in Fig. 7 to determine the probability of false state space-based detection. It can be clearly seen that, though the probability of a false detection is far greater than would be the case for a truly random sequence, it is still low enough to be insignificant.

F. Detection Parameter Values

Thus far, we have two parameters for each detection method. The goal is to set the parameters such that limit cycles are identified and removed as soon as possible, but at the same time false detections are not so frequent as to become a serious issue. The most significant parameter here is the duration of the buffer or shift register, since it determines the maximal period limit cycle that can be detected. This may be set to a small multiple of the oversampling ratio, so that it may detect limit cycles which exist in or near the bandpass region, but is still small enough

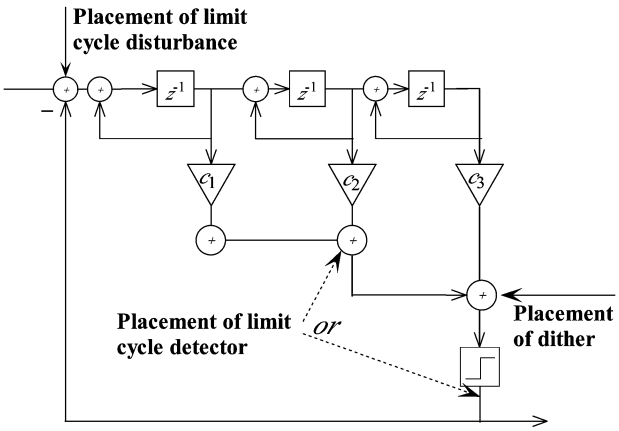

Fig. 9. A block diagrams of a third-order SDM comparing the standard placement of dither with the placement of the proposed limit cycle detector and disturbance.

that no limit cycle lasts for a significant duration before it is detected. Thus, for bitstream-based detection, $\mathrm{SR}_{\mathrm{length}}$ is typically set less than or equal to the value $P_{\max}$. The method is robust both to the size of the shift register, and the choice of when the shift register is reset.

For state space-based detection, the technique is highly robust to the tolerance T since, for SDMs of the form depicted in Fig. 1, even short term limit cycles will be detected when tolerance is set to zero. However, if resonator sections are used, then the value f may still drift in a short term limit cycle. In which case, the tolerance may be set to any value which does not yield too frequent false detections. Again, the oversampling ratio may be used. If tolerance is set to the reciprocal of a multiple of the oversampling ratio, then we may expect false detections to occur much less than once per Nyquist period.

As an example, for an oversampling ratio of 64 and sample rate of 64×44.1 kHz, we have set $P_{\max} = 256$, allowing us to detect limit cycles up to this period. Thus, the maximum time to detect any limit cycle up to period 256, using state space-based detection, from (13), is 512 samples or 0.18 ms. For bitstream-based detection, $\mathrm{SR}_{\mathrm{length}}$ may be set to 64, ensuring a probability of false detection on the order of 10^{-6} and a maximum time to detect of approximately 2 ms. In both cases, if the limit cycle is removed immediately upon detection, any artifacts resulting from the limit cycle will become negligible.

IV. Limit Cycle Removal

In [2], it was shown that, the application of dither just before the quantizer, as in Fig. 9, is a suboptimal form of limit cycle removal. This is because it has no effect on the state space variables unless it results in a change to the output bitstream. On the other hand, consider a disturbance applied at the input to the SDM. This will affect *all* state space variables. So a limit cycle may be removed by simply applying a single perturbation to the internal integrator states of a SDM. We will now show that this will always be the case for feedforward SDMs without resonator sections.

From (1), growth of the state space variables due to disturbing the input from u to $u + \varepsilon$ t time 0 is given by

$$\begin{aligned}\mathbf{s}'(n) &= \mathbf{A}^n\mathbf{s}(0) + \sum_{i=0}^{n-1}\mathbf{A}^i(u - y(n-1-i))\mathbf{d} + \mathbf{A}^{n-1}\varepsilon\mathbf{d} \\ &= \mathbf{s}(n) + \mathbf{A}^{n-1}\varepsilon\mathbf{d}. \end{aligned} \tag{15}$$

If we are in a limit cycle of period P

$$\mathbf{s}(P) = \mathbf{A}^P\mathbf{s}(0) + \sum_{i=0}^{P-1}\mathbf{A}^i(u - y(P-1-i))\mathbf{d} = \mathbf{s}(0) \tag{16}$$

and a perturbation to the input at time 0 gives

$$\mathbf{s}'(P) = \mathbf{s}(0) + \mathbf{A}^{P-1}\varepsilon\mathbf{d}. \tag{17}$$

Now, if the limit cycle had only one solution for $\mathbf{s}(0)$, then this perturbation would be guaranteed to break the limit cycle. However, a classical SDM, as described above, has a line of solutions. So the question is whether this perturbation results in a new solution to $\mathbf{s}(0)$, thus repeating the limit cycle.

From [2], all solutions to the limit cycle may be given in terms of one known solution

$$\mathbf{s}^* = \mathbf{s}(0) + \lambda\mathbf{v} \tag{18}$$

where $\mathbf{v}$ is the last column of $\mathbf{V}$ in the singular value decomposition (SVD) of $\mathbf{I} - \mathbf{A}^P$

$$\mathbf{I} - \mathbf{A}^P = U\Sigma V^T. \tag{19}$$

So the null space of $\mathbf{I} - \mathbf{A}^P$ is given by $\mathbf{v}$ where

$$(\mathbf{I} - \mathbf{A}^P)\mathbf{v} = 0. \tag{20}$$

However, notice that $\mathbf{I} - \mathbf{A}^P$ has only zero terms on the diagonal and above. That is, it is a strictly lower triangular matrix. Furthermore, all terms directly below the diagonal are nonzero. The null space can then be solved directly, regardless of P or the matrix size N.

$$\mathbf{v} = [0\,0\,\ldots\,0\,1]^T. \tag{21}$$

Comparing (20) and (21) with the formula for the growth of the perturbation, (17), we see that a limit cycle is maintained only if there exists some λ such that

$$\mathbf{A}^{P-1}\varepsilon\mathbf{d} = [0\,0\,\ldots\,0\,\lambda]^T. \tag{22}$$

It can be seen immediately that the first term of the vector on the left-hand side (LHS) of (22) is ε. Thus, the equality does not hold and any perturbation to the input must break a limit cycle.

Thus, we recommend that a small disturbance be added to the input of the SDM whenever a limit cycle has been detected.

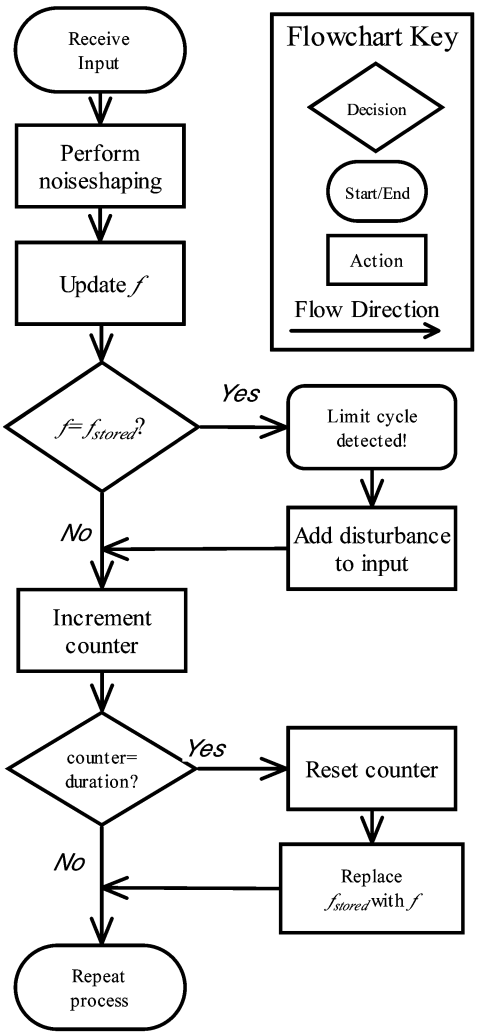

Fig. 10. A flowchart depicting the operation of a limit cycle detection and removal for an arbitrary SDM.

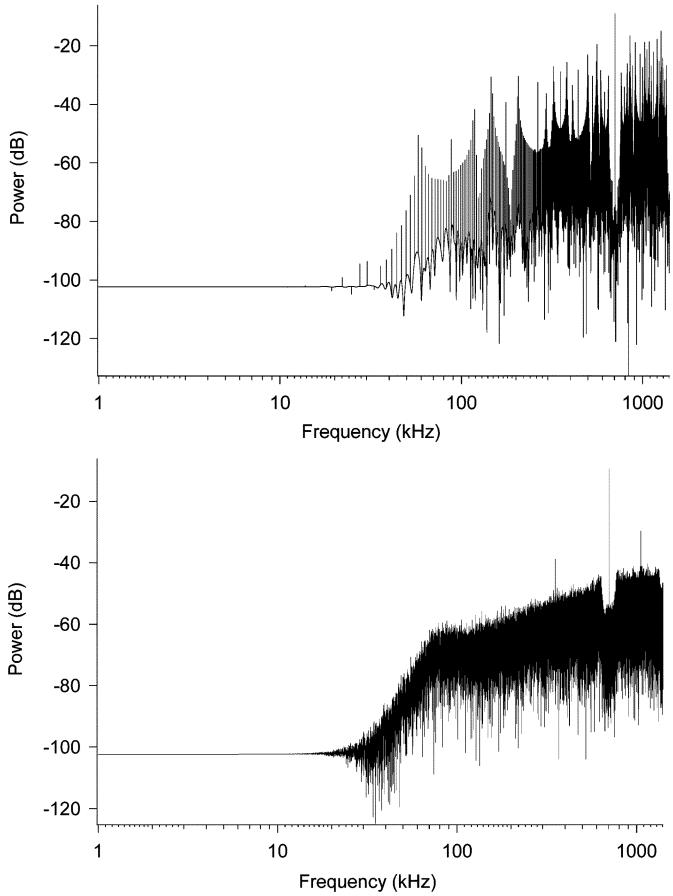

Fig. 11. The power spectrum of the output for a limit cycle (top) and after detection and removal has been implemented (bottom).

A flow chart depicting this for state space-based detection and removal is given in Fig. 10. If the buffer value f is replaced by the shift register, then this flowchart is also applicable to bitstream-based detection.

Since this modification is both minimal and guaranteed to work, it is preferable to the commonly used alternative of adding dither to the input to the quantizer. When limit cycle removal is used in conjunction with a detection method, the perturbation only needs to be added when a limit cycle has been detected. Furthermore, since the limit cycle is unstable, only a very small perturbation needs to be added. However, dithering may still be used in order to minimize noise modulation.

In Fig. 11, the power spectrum is depicted with and without limit cycle removal. The SDM with an 80-kHz corner frequency, as described in (8), is again used. Initial conditions were set to 0, input amplitude was constant at 0.5, and the sampling rate was set to 64×44.1 kHz. As can be seen, the system exhibits the expected noise shaping characteristics of a lowpass SDM, even when operating under limit cycle conditions (further discussion of the effect of limit cycles on noise shaping is provided in [10]). For limit cycle removal, the shift register length was set to 80 and a single perturbation of magnitude up to 10^{-6} was applied each time a limit cycle was identified according to the bitstream-based detection technique. For 2^{17} input samples, the perturbation is applied only 20 times. However, the sharp peaks in the power spectrum due to the occurrence of the limit cycle have been completely removed. The remaining peaks, occurring at $f_s/8$, $f_s/4$, and 3 $f_s/8$, are due to the occurrence of an idle tone and its aliasing, and remain in the spectrum independently of any limit cycles [10].

A. Disturbance Amplitude

The amplitude of the disturbance applied to the input when a limit cycle has been detected is the sole parameter used in the limit cycle removal method. The choice of the amplitude of this disturbance is a tradeoff between the reduction in SNR and stability when a large disturbance is added, and the increased amount of time it may take to destroy a limit cycle when a small disturbance is added. However, this is a minor issue since within a large range of disturbance amplitude, the disturbance has only a minimal effect on the modulator's performance and on the time it takes to destroy the limit cycle.

Fig. 12 demonstrates how the size of the perturbation affects the time it takes to destroy a limit cycle. All possible period 12 limit cycles were found, and the set of initial conditions which exactly produce each limit cycle was determined. The SDM was run with a single disturbance applied at time n, and the output

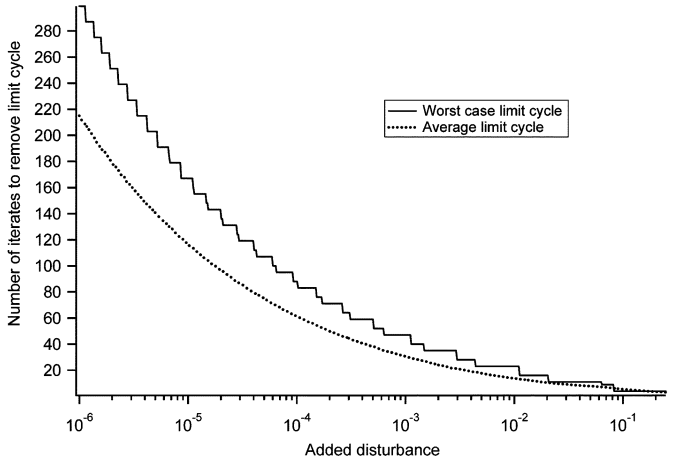

Fig. 12. For a fifth-order SDM with 100 kHz corner frequency, this depicts the number of iterates before a period 12 limit cycle is removed as a function of the size of the perturbation applied. The average limit cycle curve was found by averaging over all initial conditions which would exactly produce a limit cycle. For the worst case scenario, the initial conditions were chosen such that they would produce the most stable limit cycle possible.

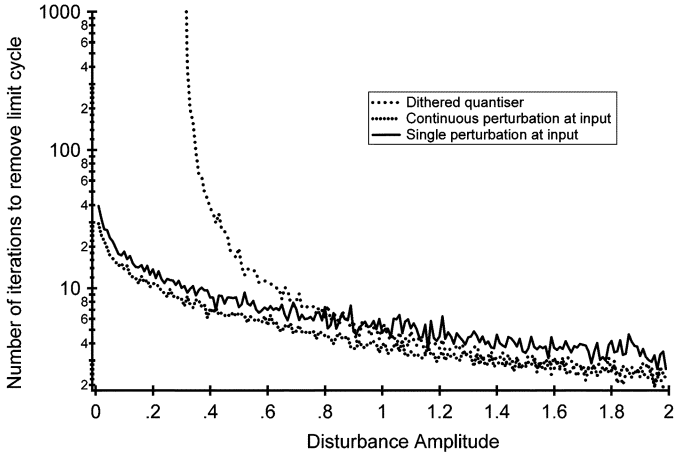

Fig. 13. The number of iterations of the SDM required to remove the largest limit cycle, as a function of the dither or perturbation width.

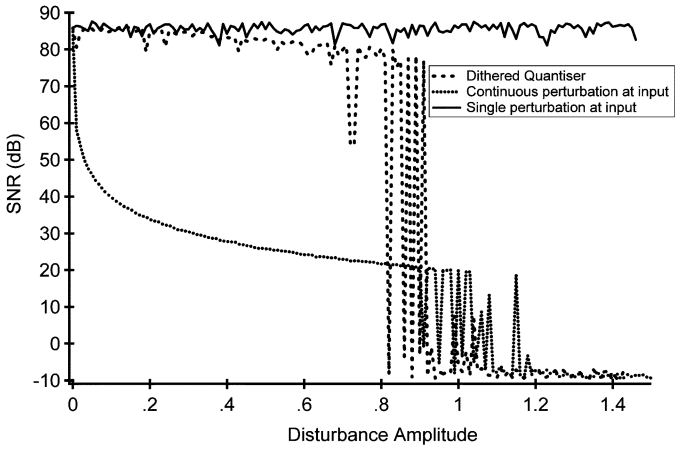

Fig. 14. SNR as a function of dither or perturbation to the input.

bitstream was monitored until it deviated from limit cycle behavior. For the average limit cycle, initial conditions producing each limit cycle were selected at random, and the number of iterates required to remove a limit cycle was averaged over all limit cycles and all selected initial conditions. For the worst case scenario, the initial conditions of the modulator were chosen such that the most stable period 12 limit cycle was produced, and such that the initial conditions were set as far as possible from those that would produce a bit flip and thus destroy the limit cycle. Nevertheless, even a disturbance on the order of 10^{-6} (akin to a change of less than -120 dB) is sufficient to eliminate the limit cycle long before it becomes problematic. The stability of limit cycles, and the growth rate of a disturbance, is discussed in detail in [2].

B. Comparison With Dither

Two aspects of the limit cycle detection and removal methods described in this paper make them a strong alternative to dithering the quantizer. First, the size of the perturbation that is required to remove a limit cycle is arbitrarily small, and hence can be made several orders of magnitude smaller than the minimum width of the dither. Second, the limit cycle removal method may be applied only when a limit cycle has been detected, as opposed to dither which is applied continuously. This is seen in Fig. 13, which depicts the number of iterations required to remove a limit cycle for dithering the quantizer and for a single perturbation applied to the input. Note also from this figure that for dither amplitude less than 0.3, dither never removes the limit cycle. The inability of low amplitude dither to remove certain limit cycles was also investigated in [2].

Since the perturbation need only be applied once, and since it can be several orders of magnitude smaller than dither, its effect on the SNR is negligible. In order to provide a meaningful depiction of the comparison between SNRs for the limit cycle removal methods, we also consider continuously perturbing the input. At large amplitude disturbance, this is equivalent to dithering the input signal. This is not recommended since it results in significant input noise that is not moved away from the passband.

The SNR as a function of the disturbance amplitude is depicted in Fig. 14, where a sinusoidal input with amplitude 0.5 and period of 101 samples (approximately 2.17 kHz for a sampling rate of 64×44.1 kHz) was chosen as input. Values of SNR close to or below zero are indicative of unstable behavior. This agrees with previous work which noted that full scale dither, whether of triangular or rectangular distribution, will overload the quantizer in a 1–bit SDM [18] and result in instability. Thus, smaller amplitude dither is recommended for one bit SDMs [19], and this is the justification for analysis of SNR, stability, and time to remove a limit cycle as a function of dither amplitude.

As can be seen in Fig. 14, applying a single disturbance to the input yields no noticeable drop in SNR. Dithering the quantizer degrades the SNR slightly as can be established by theory [6]. Applying a continuous disturbance to the input leads to a rapid drop in the SNR. However, this is misleading since, as mentioned previously, the disturbance need only be applied rarely, regardless of whether the limit cycle detection method is used, and the disturbance may be made extremely small.

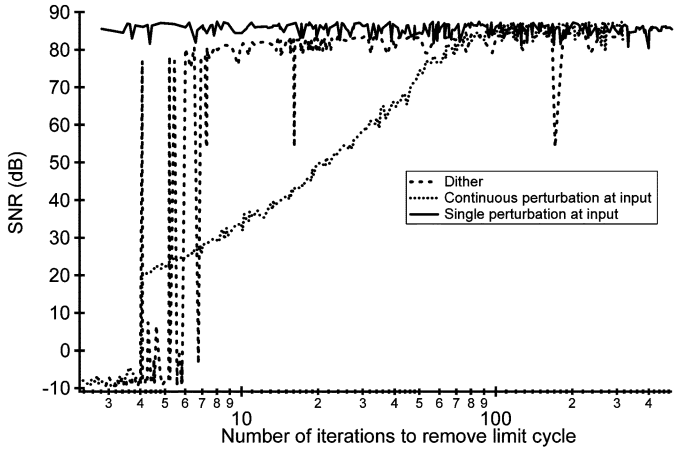

Fig. 15. SNR as a function of the number of iterations required to remove the largest period 12 limit cycle for different limit cycle removal methods.

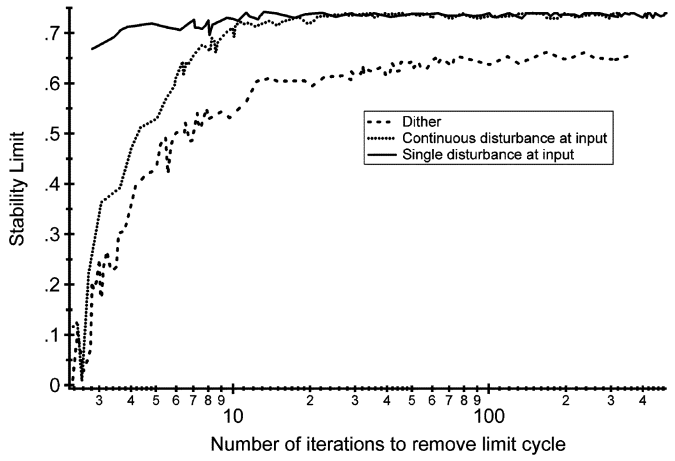

Fig. 16. Stability as a function of the number of iterations required to remove the largest period 12 limit cycle for different limit cycle removal methods.

Thus, to investigate the behavior on a more realistic level, we consider the SNR as a function of the time required to remove a limit cycle, as shown in Fig. 15. When a disturbance on the order of 10^{-3} is continually applied to the input, corresponding to roughly 75 iterations required to remove the limit cycle, then the SNR performance is equivalent to dithering the quantizer. The performance decays rapidly for large continuous perturbations. However, when a small disturbance is applied, or when it is applied infrequently (such as only when a limit cycle is detected), then this technique yields higher SNR performance than dithering.

Similar behavior can be seen in Fig. 16 which depicts the largest possible constant input that will yield stable behavior as a function of the number if iterations required to remove the largest period 12 limit cycle, which in turn is a function of the amplitude of the dither or disturbance applied. In this case, any form of disturbing the input yields better stability performance than dithering the quantizer. This is because when the input signal is disturbed, the output is still an accurate representation of the recent inputs to the quantizer. However, dithering the quantizer intentionally results in an output quantization that does not represent recent inputs, and when this (incorrect) quantization value is fed back to the input it pushes the integrator states closer to instability.

V. Implementation

A. Implementation in Low-Order SDMs

Thus far, we have concentrated on fifth-order SDM designs. The procedures for limit cycle detection and removal outlined in Sections III and IV are independent of the SDM order with one notable exception. For a first-order SDM, any constant rational input results in a limit cycle, independent of the integrator states [9]. Hence, a perturbation to the input, even if it breaks the limit cycle, will result in the SDM immediately returning to that same limit cycle. Therefore, there is no noticeable benefit to implementation of this technique in first order SDMs.

B. Circuit Implementation

The method of detection and removal of limit cycles can be implemented in digital or analog circuitry. Detection of limit cycles is performed either on the output bitstream (wholly digital—for both ADCs and DACs) or on the internal states of the SDM (digitally in a DAC, or analog in an ADC).

For analog detection using internal states, a single voltage value, which is already available internally, is stored. This can be stored on a capacitor. Leakage is not a major issue since the value is stored for short duration. A simple, low bit counter is used to determine when this value is updated. This stored value is compared with a new value at each iterate. The comparison may be made with a simple window comparator.

For digital detection using internal states, a single storage register is used, which need only be (at most) up to the internal resolution of the SDM. Again, a counter and logic gate are deployed, as with the analog implementation.

Limit cycle detection using the output bitstream is a digital process. It requires the implementation of 2 shift registers storing output bits and as with analog detection, a low bit counter. Bit comparisons are performed in parallel and a multi-input AND gate is used to test equivalence of shift registers.

For removal of limit cycles in ADC, almost any analog source capable of producing low voltage output may be deployed. For instance, one may use a zener diode noise source. Other simple implementations are available. For removal in a DAC, a low complexity pseudorandom number generator may be used to modify the least significant bits of a signal. If deployed only when limit cycles are detected, a logic gate is used to turn it on. There are no requirements that the disturbance be close to ideal noise, and simply flipping the least significant bit may suffice and will result in the smallest disturbance possible.

Note that all methods of detection are simple to implement, operate at the sampling frequency, and do not require major modifications to the design of an SDM. Additional hardware complexity is very small, and far less than other control methods which improve the performance of SDMs [20].

VI. Conclusion

In this paper, we have described some of the properties of limit cycle detection and removal schemes which have been implemented for feedforward SDMs. These methods successfully detect any limit cycle up to a given period and may remove any limit cycle. They have a very low probability of false detection, are easy to implement, and outperform current techniques such as dithering. It is hoped that these methods may soon be realized in commercial SDMs.

The state space-based limit cycle detection method relies on calculation of a single scalar quantity. It allows one to find short-term limit cycles. This quantity (10) is easily found from the operation of the SDM and the decision to ignore s_N allows us to find all limit cycles that occur up to a given period, as well as short term limit cycles which repeat for only a small number of periods. It is robust to the choice of parameter settings, and may detect limit cycles which occur at any time during the operation of the SDM, Furthermore, it is independent of the order of the SDM, its noise shaping characteristics, and independent of the input to the SDM and its initial conditions. The mechanism of limit cycle detection operates at the speed of the SDM, and may be used in tandem with any limit cycle removal method.

The output bitstream-based limit cycle detection method, using only two shift registers, will find all possible limit cycles up to a given period which may be larger than the shift register length). It has all the advantages of the state space-based method. Furthermore, it is robust against any additional parameters (such as shift register length) and easy to implement.

Either of the two limit cycle detection methods may be combined with the limit cycle removal method to yield a highly effective technique for limit cycle removal which does not require continual dithering.

The limit cycle removal method may be applied continuously, or only when a limit cycle has been detected. If used in tandem with a limit cycle detection mechanism, it is independent of the choice of detection method. It is guaranteed to remove any limit cycle, and is independent of SDM characteristics. Its effect on the SDM (other than removing limit cycles) is minimal. It is robust to the choice of parameter settings, such as the size of the disturbance and the exact moment at which it is applied.

These methods may also be applied to other SDM designs. For instance, a popular alternative design to the feedforward, SDM, is the feedback SDM. This is often used when a superior antialiasing effect of the signal transfer function is required [13], [21]. For both designs, the placement of the transition matrix $\mathbf{A}$ in the state space equations is identical. However, for feedback designs, the coefficient vector acts as a constant that is added or subtracted from the state space variables every iteration. Here, short period limit cycles are very rare. Minor modifications must be made in limit cycle detection and removal mechanisms. For state variable-based detection, we do not need to identify limit cycles using anything other than the quantizer input. For bitstream-based detection, the algorithm is exactly the same as that described in Section III-B. Limit cycle removal may be implemented by adding a disturbance either to the system input, or to the system output just prior to feedback. Detailed discussions of how to apply limit cycle detection and removal to this and other alternative SDM designs, and practical circuit realizations of these methods, remain topics for future work.

References

[1] S. Ouzounov, H. Hegt, and A. van Roermund, "Sigma-delta modulators operating at a limit cycle," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 53, no. 2, pp. 399–403, Feb. 2006.

[2] D. Reefman, J. Reiss, E. Janssen, and M. Sandler, "Description of limit cycles in sigma-delta modulators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, pp. 1211–1223, 2005.

[3] S. Yamaki, M. Abe, and M. Kawamata, "On the absence of limit cycles in state-space digital filters with minimum l2-sensitivity," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 1, pp. 46–50, Jan. 2008.

[4] S. P. Lipshitz and J. Vanderkooy, "Towards a better understanding of 1-bit sigma-delta modulators," in *Proc. Audio Eng. Soc. 110th Convention*, Amsterdam, Holland, 2001.

[5] H. Hsieh and C.-L. Lin, "Spectral shaping of dithered quantization errors in sigma-delta modulators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 54, pp. 974–980, 2007.

[6] R. A. Wannamaker, S. P. Lipshitz, J. Vanderkooy, and J. N. Wright, "A theory of non-subtractive dither," *IEEE Trans. Signal Process.*, vol. 48, pp. 499–516, 2000.

[7] S. I. Mann and D. P. Taylor, "Limit cycle behavior in the double-loop bandpass sigma delta A/D converter," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 46, no. 8, pp. 1086–1089, Aug. 1999.

[8] N. Bridgett and C. P. Lewis, "Effect of initial conditions on limit cycle performance of second order sampled data sigma delta modulator," *Electron. Lett.*, vol. 26, pp. 817–819, 1990.

[9] V. Friedman, "The structure of limit cycles in sigma delta modulation," *IEEE Trans. Commun.*, vol. 36, pp. 972–979, 1988.

[10] E. Perez Gonzalez and J. D. Reiss, "Idle tone behavior in sigma delta modulation," in *Proc. 122nd Audio Eng. Soc. Convention*, Vienna, Austria, 2007.

[11] H. Huijberts, A. Pavlov, and J. D. Reiss, "Boundedness and aperiodicity of commercial sigma delta modulators," in *Proc. 1st IFAC Conf. Anal. Control of Chaotic Syst. (CHAOS'06)*, Reims, France, 2006.

[12] J. D. Reiss and M. Sandler, "The harmonic content of a limit cycle in a DSD bitstream," in *Proc. Audio Eng. Soc. 116th Convention*, Berlin, Germany, 2004.

[13] D. Reefman, J. D. Reiss, E. Janssen, and M. Sandler, "Description of limit cycles in feedback sigma delta modulators," in *Proc. Audio Eng. Soc. 117th Convention*, San Francisco, CA, 2004.

[14] J. D. Reiss and M. B. Sandler, "A mechanism for the detection and removal of limit cycles in the operation of sigma delta modulators," U.K. British Patent Appl. No. 0514677.4, Jul. 18, 2005.

[15] D. Hyun and G. Fischer, "Limit cycles and pattern noise in single-stage single-bit delta-sigma modulators," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 49, pp. 646–656, 2002.

[16] J. A. S. Angus, "The effect of idle tone structure on effective dither in delta-sigma modulation systems," in *Proc. Audio Eng. Soc. 112th Convention*, Munich, Germany, 2002.

[17] Super Audio CD Audio Signal Properties (SACD Scarlet Book) Mar. 2003 [Online]. Available: www.superaudiocd.philips.com/Assets/Downloadablefile/SACD_Au_DE_130(scarlet-book)-2325.pdf, Sony and Philips

[18] S. P. Lipshitz and J. Vanderkooy, "Why professional 1-bit sigma-delta conversion is a bad idea," in *Proc. Audio Eng. Soc. 109th Convention*, Los Angeles, CA, Sep. 2000, pp. 22–25.

[19] D. Reefman and P. Nuijten, "Why direct stream digital (DSD) is the best choice as a digital audio format," in *Proc. Audio Eng. Soc. 110th Convention*, Amsterdam, The Netherlands, 2001.

[20] C. Y.-F. Ho, B. W.-K. Ling, and J. D. Reiss, "Fuzzy impulsive control of high order interpolative lowpass sigma delta modulators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 53, no. 10, pp. 2224–2233, Oct. 2006.

[21] S. Norsworthy, R. Schreier, and G. Temes, *Delta-Sigma Data Converters*. Piscataway, NJ: IEEE, 1997.

Joshua D. Reiss was born in 1971. He received the B.S. degrees in both physics and mathematics, and the Ph.D. degree in physics from the Georgia Institute of Technology, Atlanta, specializing in the analysis of chaotic time series.

He is a Lecturer with the Centre for Digital Music and the Digital Signal Processing Group, Electronic Engineering Department, Queen Mary, University of London, London, U.K. In June 2000, he joined the Audio Signal Processing Research Lab at King's College, London, and moved to Queen Mary in 2001. He made the transition from chaos theory to audio processing through his work on sigma delta modulators, which has lead to a nomination for a best paper award from the IEEE, as well as a U.K. patent. His work also includes significant contributions to the fields of music retrieval and processing, audio effects, satellite navigation, and nonlinear dynamics.

Dr. Reiss is a member of the Audio Engineering Society (AES), as well as a member of the Review Board for the *Journal of the AES* and Vice-Chair of the Technical Committee on High-Resolution Audio. He was on the Organizing Committee of DAFx2003 and MIREX2005, and was recently Programme Chair for the 2005 International Conference on Music Information Retrieval and General Chair of the 2007 AES Conference on High Resolution Audio. As coordinator of the EASAIER project, he leads an international consortium of seven partners working to improve access to sound archives in museums, libraries, and cultural heritage institutions.

Mark Sandler (SM'98) was born in 1955. He received the B.Sc. and Ph.D. degrees from the University of Essex, Essex, U.K., in 1978 and 1984, respectively.

He joined Queen Mary, University of London, London, U.K., in 2001, as a Professor of Signal Processing and a Director of the Centre for Digital Music, after 19 years with King's College London. He was Founder and CEO of Insonify, Ltd., London, an Internet audio streaming startup for 18 months. He has published more than 250 papers in journals and conferences.

Dr. Sandler is a Fellow of the Institute of Electrical Engineers (IEE) and a Fellow of the Audio Engineering Society. He is a two-time recipient of the IEE A. H. Reeves Premium Prize.