Look-ahead Sigma-Delta Modulation and its application to Super Audio CD

Erwin Janssen

The work described in this thesis has been carried out at the Philips Research Laboratories and NXP Semiconductors, Eindhoven, the Netherlands, as part of the Philips/NXP research program.

Janssen, E.

Look-ahead Sigma-Delta Modulation
and its application to Super Audio CD

Proefschrift Technische Universiteit Eindhoven, 2010

Trefwoorden: 1-bit audio, digital-to-digital conversion, linearization, look-ahead, noise shaping, sigma-delta modulation, signal processing

A catalogue record is available from the Eindhoven University of Technology Library
ISBN: 978-90-386-2364-1

Look-ahead Sigma-Delta Modulation and its application to Super Audio CD

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
rector magnificus, prof.dr.ir. C.J. van Duijn, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op woensdag 1 december 2010 om 16.00 uur

door

Erwin Janssen

geboren te Ede

Dit proefschrift is goedgekeurd door promotor:

prof.dr.ir. A.H.M. van Roermund

Samenstelling promotiecommissie:

prof.dr.ir. A.H.M. van Roermund	Technische Universiteit Eindhoven
prof.dr.ir. A.C.P.M. Backx	Technische Universiteit Eindhoven
dr.ir. J.A. Hegt	Technische Universiteit Eindhoven
dr.ir. P.C.W. Sommen	Technische Universiteit Eindhoven
prof.dr.ir. B. Nauta	Universiteit Twente
prof.dr.ir. G. Gielen	Katholieke Universiteit Leuven
dr. D. Reefman	Philips Research
dr.ir. L.J. Breems	NXP Semiconductors

Contents

List of symbols and abbreviations

t_0	current time step
AC	alternating current
AD	analog to digital
ADC	analog to digital converter
C	accumulated cost value or cost function
c	cost value
DA	digital to analog
DAC	digital to analog converter
dBFS	decibels full scale
DC	direct current, 0 Hz
DD	digital to digital
DDC	digital to digital converter
DSD	Direct Stream Digital
DSM	Delta-Sigma Modulator
DST	Direct Stream Transfer
ENOB	effective number of bits
ERBW	effective resolution bandwidth
ETSDM	Efficient Trellis SDM
FB	feed-back
FF	feed-forward
FoM	figure of merit
Fs	sampling rate
HD	harmonic distortion
HMM	hidden Markov model
L	the number of bits latency, or the trace-back depth in bits, for all the different look-ahead algorithms
LA	look-ahead
LASDM	look-ahead SDM
M	the number of parallel paths in the Pruned Tree sigma-

	delta modulation algorithm (for SA-CD)
N	the number of bits over which uniqueness of the parallel solutions is determined in the (Efficient) Trellis sigma-delta modulation algorithm. Also determines the number of parallel solutions (2^N) for the Trellis algorithm.
NS	noise-shaping
NTF	noise transfer function
OSR	oversampling ratio
PCM	pulse-code modulated
PDF	probability density function
PDM	pulse density modulation
PTSDM	Pruned Tree SDM
PWM	pulse width modulation
SA-CD	Super Audio CD
SD	Sigma-Delta
SDM	Sigma-Delta Modulator
SFDR	spurious free dynamic range
SINAD	signal to noise and distortion ratio
SNDR	SINAD
SNR	signal to noise ratio
STF	signal transfer function
TD	time domain
THD	total harmonic distortion
TSDM	Trellis SDM

Chapter 1

Introduction

In March 1999 the Super Audio Compact Disc (Super Audio CD, SA-CD), the successor of the normal audio CD, was presented to the world. This new audio carrier, conceived by Philips and Sony, makes use of a radically new way to store and reproduce audio signals. Instead of working with the traditional 44.1 kHz sampling rate and 16-bit pulse-code modulated (PCM) signals, a 2.8 MHz 1-bit format is used to store the audio signal. The new format is marketed to deliver a signal-to-noise ratio (SNR) of 120 dB and a signal bandwidth of 90 kHz, as opposed to an SNR of 96 dB and a bandwidth of 20 kHz for the normal audio CD. The decision for this alternate encoding format was made years earlier, when 1-bit Analog-to-Digital (AD) audio Sigma-Delta (SD) converters were still delivering the highest signal conversion quality. In fact, virtually all of the high quality AD and digital-to-analog (DA) converters that were used at that time for the generation and reproduction of CD quality PCM audio were based on 1-bit converters. It was reasoned that a higher audio quality could be obtained by removing the decimation and interpolation filters that performed the conversion from 1-bit to PCM and vice versa, and by storing the 1-bit signal from the Sigma-Delta Modulator (SDM) directly on the disc.

Although the idea of storing the 1-bit SDM output signal directly on the disc sounds very reasonable, in practice things work differently, and the original recorded signal is never stored directly on a disc. As a result, there is a clear need for high quality digital 1-bit Sigma-Delta Modulators that generate bitstreams that have a high lossless compression gain, as will be explained in the next section. After the motivation for the work, the aims and the scope of the thesis are presented. Finally, a short description of the contents of each chapter of the thesis is given.

1.1 Motivation

In the process of recording an SA-CD, typically, a number of recordings of the same performance are made, and at a later stage in the studio those recordings are edited and processed, e.g. removal of coughs from an audience or the equalization of the audio levels, until the desired sound quality is obtained. This process of editing and processing can only be performed on multi-bit (PCM) signals, and only once all this work is done the 1-bit signal that will be stored on the SA-CD disc will be generated. Thus, if it is assumed that all the digital processing on the audio signal is without any loss of the signal quality, the final signal quality of the 1-bit signal that is stored on the disc is determined by the initial analog-to-digital conversion and the final digital-to-digital (DD) conversion.

Nowadays, the highest quality analog-to-digital conversion for audio applications is obtained with a multi-bit SDM. Such a converter can deliver a very high SNR and very low distortion levels. From the output of the SDM a PCM signal is generated, but now with a higher resolution and much higher sampling rate than what is used for CD. After all the processing on the multi-bit signal is performed, the final 1-bit signal is generated. Traditionally, this is done with a digital 1-bit SDM. However, with a normal SDM it is not trivial to generate a 1-bit signal with the desired ultra-high quality under all signal conditions. For example, for extremely high signal levels a 1-bit SDM can generate significant distortion, especially if the modulator is designed to deliver a very high SNR for normal signal levels. Besides this potential signal quality issue there is a much bigger issue that, with traditional sigma-delta modulation approaches, can not be solved without jeopardizing the signal quality: the risk of not realizing a long enough playback duration.

The SA-CD standard supports, in addition to a normal stereo recording, also the possibility to store a multi-channel version of the same recording. In order to fit all the data on the 4.7 gigabyte disc and obtain a playback duration of at least 74 minutes, the standard playback duration of the normal audio CD, lossless data compression is applied to the 1-bit audio signal. Only if the compression gain, the ratio that indicates the amount of data size reduction, is high enough it will be possible to obtain the required 74 minutes of playback time. Since the data compression algorithm is lossless, the compression gain depends on the redundancy in the 1-bit encoded audio signal, and this can only be influenced with the SDM design. However, the only solution to increase the redundancy is to reduce the signal conversion quality of the SDM,

and since SA-CD is about delivering high audio quality this is not an acceptable solution.

From the above it is clear that there is a strong motivation to realize a 1-bit digital sigma-delta modulation solution that is able to realize a very high signal conversion quality and that is simultaneously able to generate bitstreams that are compatible with the SA-CD lossless data compression algorithm. As demonstrated by Kato in [37, 38] the use of a look-ahead modulator instead of a normal SDM can bring significant improvements to the signal conversion quality. Although the computational load of his solution is too large for the approach to be practically usable, it does provide a good starting point for the exploration of alternative look-ahead approaches that are able to improve the signal conversion quality at a reasonable computational cost.

1.2 Aim of the thesis

The aim of this thesis is to expand and improve upon the existing knowledge on discrete-time 1-bit look-ahead sigma-delta modulation in general, and to come to a solution for the above mentioned specific issues arising from 1-bit sigma-delta modulation for SA-CD.

In order to achieve this objective an analysis is made of the possibilities for improving the performance of digital noise shaping look-ahead solutions. In this context "performance" has a broad definition and encompasses the standard signal-to-noise ratio and linearity performance indicators, the 1-bit SDM specific measures of stability and noise modulation, and also the computational load associated with a look-ahead algorithm. In the specific case of a look-ahead modulator for SA-CD also the lossless compression gain that is obtained on the output bitstream is evaluated.

On the basis of the insights obtained from the analysis, several novel generic 1-bit look-ahead solutions that improve upon the state-of-the-art will be derived and their performance will be evaluated and compared. Finally, all the insights are combined with the knowledge of the SA-CD lossless data compression algorithm to come to a specifically for SA-CD optimized look-ahead design.

1.3 Scope of the thesis

Almost all of the work described in this thesis has a general focus on (digital) 1-bit look-ahead sigma-delta modulation, and is independent of the sampling rate and the loop filter type of the converter. However, since the possibilities of look-ahead modulation are investigated with a Super Audio CD application in mind, the SDM design parameters used throughout the work are selected in line with the Super Audio CD standard. This translates to a sampling rate for the studied Sigma-Delta Modulators of 64·44.1 kHz, approximately 2.8 MHz, and the use of interpolative (low-pass) loop filters. The SNR, the signal-to-noise-and-distortion ratio (SINAD), the total harmonic distortion (THD), and the spurious free dynamic range (SFDR), are always evaluated over the audio bandwidth of 20 kHz. Besides these SA-CD specific parameters no use is made of any SA-CD specific nomenclature, except for chapter 10 where a minimal amount of usage can not be avoided. All the demonstrated SDM implementations have been realized in software, i.e. written in ANSI C, and make use of floating point arithmetic. Only limited attention is paid to the challenges of realizing a hardware solution, since in the context of Super Audio CD the primary intended use is in a software application.

1.4 Organization of the thesis

In chapter 2, a basic introduction to sigma-delta modulation and the performance evaluation of Sigma-Delta Modulators is given. Readers familiar with traditional sigma-delta modulation for AD and DD conversion and the possible artifacts resulting from 1-bit sigma-delta modulation can skip this chapter and immediately continue with chapter 3.

Traditionally, signal conversion quality is characterized with steady-state signals. In the case of a linear data converter this procedure will also give the performance for non-steady-state signals. However, since a 1-bit SDM is a non-linear data converter, it is not guaranteed that the steady-state performance is representative for non-steady-state signals. In chapter 3 this potential discrepancy is investigated.

In chapter 4, a generic model of a noise-shaping quantizer is derived. This model is subsequently used in chapter 5 to come to a noise-shaping quantizer model for a look-ahead converter. Next, the main look-ahead principles are introduced, accompanied with an analysis of the benefits and disadvantages. The basic full look-ahead algorithm is presented,

and an analysis is made of the possibilities for realizing a look-ahead enabled AD converter. Although this idea is rejected, it is clear that large benefits can be expected from look-ahead based DD conversion, but only if an approach with a reduced computational load can be realized.

The possibilities for reducing the computational load of the full look-ahead algorithm for DD conversion are investigated in chapter 6. Since the obtainable reduction is rather limited, an alternative approach, i.e. pruning of the solution space, is investigated. It is concluded that, with a proper pruning algorithm, it should be possible to realize solutions that result in large computational savings and that have a limited impact on the obtainable signal conversion performance. Therefore, the next chapters focus on pruned look-ahead algorithms.

In chapter 7, an analysis is made of the Trellis sigma-delta modulation algorithm by Kato. An improvement of the signal conversion quality, compared to a normal SDM, is realized but at a very large computational cost.

Further analysis of the Trellis sigma-delta modulation algorithm in chapter 8 reveals that only a fraction of all the parallel solutions contributes to the final output. The Efficient Trellis sigma-delta modulation algorithm makes use of this observation and prunes the solution space further, thereby enabling a larger pruned look-ahead depth that results in an improvement of the signal conversion quality, as well as a reduction in the computational load.

In chapter 9, the Pruned Tree sigma-delta modulation algorithm, that is an improvement over the Efficient Trellis sigma-delta modulation algorithm, is discussed. The pruning criteria that is applied in the Efficient Trellis sigma-delta modulation algorithm is effective for reducing the number of parallel solutions, but also adds a significant computational overhead to the algorithm. By changing the initial conditions of the look-ahead modulator the pruning criteria can be relaxed, which results in a computationally more efficient solution that, typically, delivers performance that is on par with that of the Efficient Trellis sigma-delta modulation algorithm, but that is sometimes even better.

In the Pruned Tree sigma-delta modulation algorithm for SA-CD, described in chapter 10, a cost function is added to the original Pruned Tree sigma-delta modulation algorithm that reflects the predictability of the output bitstream. This addition results in a dual optimization that takes both the signal quality into account and improves the lossless data compression gain of the output signal.

In chapter 11, a comparison is made between the various look-ahead

techniques that are detailed in the previous chapters. This comparison includes an analysis of the algorithmic differences, and a comparison of the functional performance.

In the previous chapters it was found that there appears to be a limit on the SNR that can be achieved with a fifth order 1-bit SDM. In chapter 12 this phenomenon is analyzed in detail and new results on the limits of 1-bit noise shaping are presented.

Finally, in chapter 13 the general conclusions on the work described in this thesis are presented.

Chapter 2

Basics of sigma-delta modulation

The principle of sigma-delta modulation, although widely used nowadays, was developed over a time span of more than 25 years. Initially the concept of oversampling and noise shaping was not known and the search for an efficient technique for transmitting voice signals digitally resulted in the Delta Modulator. Delta modulation was independently invented at the ITT Laboratories by Deloraine et. al [12,13] the Philips Research Laboratories by de Jager [11], and at Bell Telephone Labs [9] by Cutler. In 1954 the concept of oversampling and noise shaping was introduced and patented by Cutler [10]. His objective was not to reduce the data rate of the signal to transmit as in earlier published work, but to achieve a higher signal-to-noise ratio in a limited frequency band. All the elements of modern sigma-delta modulation are present in his invention, except for the digital decimation filter required for obtaining a Nyquist rate signal. The name Delta-Sigma Modulator (DSM) was finally introduced in 1962 by Inose et al. [26,27] in their papers discussing 1-bit converters. By 1969 the realization of a digital decimation filter was feasible and described in a publication by Goodman [17]. In 1974 Candy published the first complete multi-bit Sigma-Delta Modulator (SDM) in [7]. Around the same time the name SDM was introduced as an alternative for Delta-Sigma Modulator and since then both names are in use. In this thesis the oversampled noise-shaping structure will be referred to as SDM. According to the author SDM is the more appropriate name since the integration or summing (the sigma) is over the difference (the delta).

In the 70's, because of the initially limited performance of Sigma-Delta

Modulators, their main use was in encoding low frequency audio signals (analog-to-digital conversion) using a 1-bit quantizer and a first or a second order loop filter. The creation of black and white images for print from a gray scale input was another application where Sigma-Delta noise-shaping techniques were used (digital-to-digital conversion). Since then a lot of research on improving SDM performance has been performed and great improvements have been realized. Nowadays top of the line SDM based analog-to-digital converters (ADCs) use a multi-bit quantizer and a high-order loop filter and are capable of converting 10's of MHz of bandwidth with high dynamic range. Because of high power efficiency, Sigma-Delta based analog-to-digital converters are used in the radio of mobile telephones. Another example of the efficient use of sigma-delta modulation techniques is the Super Audio CD format which uses a 64 times oversampled 1-bit signal for delivering a 120 dB signal-to-noise ratio (SNR) over the 0-20 kHz band. In this specific example the decimation filter is omitted and the oversampled signal is directly stored as to minimize signal operations and therefore maximize the signal quality. An omnipresent example of sigma-delta modulation in digital-to-analog conversion can be found in portable audio playback devices, e.g. IPOD and MP3 players. The audio digital-to-analog converter (DAC) in these devices realizes its performance using noise shaping (NS) and pulse-width-modulation (PWM) or pulse-density-modulation (PDM) techniques. These PWM/PDM signals are typically generated using a (modified) digital SDM.

Although all these SDM solutions are optimized for a certain application and context, they still share the same underlying basic principles of oversampling and noise shaping. Oversampling is the process of taking more samples per second than required on the basis of the Nyquist-Shannon criterion. By changing the sampling rate the signal power and total quantization noise power is not affected. Therefore, the signal to quantization noise ratio is not changed. However, the quantization noise is spread over a larger frequency range, reducing the spectral density of the quantization noise. If now only the original Nyquist band is considered, the quantization noise power is reduced by 3 dB for every doubling of the oversampling ratio and the signal to quantization noise ratio is improved accordingly. This effect is illustrated in fig. 2.1 for an oversampling ratio (OSR) of 1, 2, and 4 times.

Noise shaping is applied as a second step to improve the signal to quantization noise ratio. In this process the frequency distribution of the quantization noise is altered such that the quantization noise density reduces in the signal band. As a result the noise density increases at other frequencies where the noise is less harmful. This effect is depic-

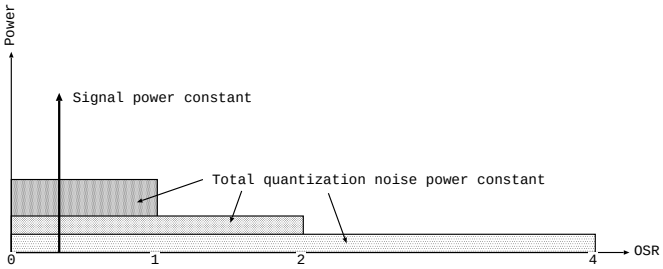


Figure 2.1: Oversampling does not affect the signal power or total quantization noise power but reduces the noise spectral density.

ted in fig. 2.2, where low frequency noise is pushed to high frequencies. The amount of quantization noise is not changed by this process but the signal to noise ratio is increased in the low frequency area of the spectrum. In an SDM the techniques of oversampling and noise shaping are combined, resulting in an increased efficiency since now the quantization noise can be pushed to frequencies far from the signal band.

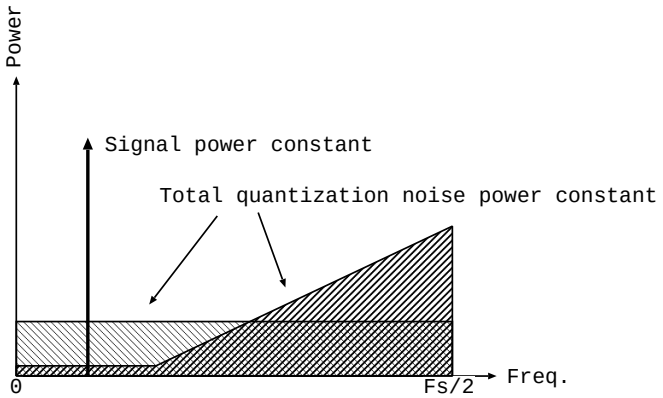


Figure 2.2: Low frequency noise is pushed to high frequencies by noise shaping.

All SDM structures realize the shaping of noise with an error minimizing feedback loop in which the input signal x is compared with the quantized output signal y, as depicted in fig. 2.3. The difference between these two signals is frequency weighed with the loop filter. Differences between the

input and output that fall in the signal band are passed to the output without attenuation, out-of-band differences are suppressed by the filter. The result of the weighing is passed to the quantizer, which generates the next output value y. The output y is also fed back to the input, to be used in the next comparison. The result of this strategy is a close match of input signal and quantized output in the pass-band of the filter, and shaping of the quantization errors such that those fall outside the signal band.

Figure 2.3: Generic model of the Sigma-Delta noise-shaping loop, consisting of 2-input loop filter and quantizer.

In sec. 2.1 the noise-shaping loop in data converters will be examined in detail, revealing that in reality only analog-to-digital (AD) and digital-to-digital (DD) noise shaping conversion exists. Over the last decennia a great variety of noise-shaping loops have been developed, but all originate from a minimal number of fundamental approaches. The most commonly used configurations are discussed in sec. 2.2. During the design phase of an SDM the noise-shaping transfer function is typically evaluated using a linear model. In reality, especially for a 1-bit quantizer, the noise transfer is highly non-linear and large differences between predicted and actual realized transfer can occur. In sec. 2.3 the linear modeling of an SDM is examined and it will be shown that simulations instead of calculations are required for evaluating SDM performance. Several criteria exist for evaluating the performance of an SDM. The criteria can be differentiated between those that are generic and are used for characterizing data converters in general, and those that are only applicable for Sigma-Delta converters. Both types are discussed in sec. 2.4.

2.1 AD, DD, and DA Sigma-Delta conversion

2.1.1 AD conversion

The most well-known form of sigma-delta modulation is analog-to-digital conversion. In fig. 2.4 the main building blocks of a generic Sigma-Delta ADC are shown. In the figure the analog and digital domains are indicated as well. The analog signal that will be converted, as well as the DAC feedback signal, enter the analog loop filter at the left side of the figure. The output of the loop filter is converted to an n-bit digital signal by the quantizer (ADC). This n-bit digital signal is passed to a digital decimation filter and to the feedback DAC. The decimation filter removes the out-of-band quantization noise, thereby converting the high rate low resolution signal to a high resolution low rate signal. The feedback DAC performs the inverse function of the ADC (quantizer) and converts the n-bit digital code to an analog voltage or current, closing the Sigma-Delta loop.

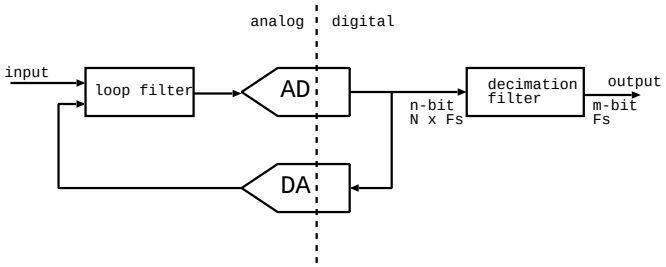


Figure 2.4: Main building blocks of a Sigma-Delta analog-to-digital converter.

Several different types of analog Sigma-Delta Modulators exist, varying in for example the way the loop filter is functioning (e.g. continuous time or discrete time) or how the DAC is constructed (e.g. switched capacitor or resistor based). Independent of these details, in all structures the use of a low resolution ADC and DAC is key. The coarse quantization results in a large amount of quantization noise which is pushed out of band by the loop filter. The number of bits used in the ADC and DAC is typically in the range 1-5. A 1-bit quantizer is easier to build than a 5-bit quantizer, requires less area and power, and is intrinsically linear, but has the disadvantage that less efficient noise shaping can be realized and that a higher oversampling ratio is required to compensate for this.

The final Sigma-Delta output, i.e. at the output of the decimation filter, will be an m-bit word where m can be as high as 24. The number of bits is independent on the number of bits used in the internal ADC and DAC. Sometimes only the part before the decimation filter is considered in discussions about Sigma-Delta Modulators.

2.1.2 DD conversion

In a digital-to-digital Sigma-Delta converter an n-bit digital input is converted to an m-bit digital output, where n is larger than m. The sampling rate of the signal is increased during this process in order to generate additional spectral space for the quantization noise. The main building blocks of a generic DD SDM are shown in fig. 2.5. The n-bit signal is first upsampled from Fs to N x Fs in the upsampling filter. The resulting signal is passed to the actual SDM loop. This loop is very similar to the one in fig. 2.4, except that now everything is in the digital domain. The ADC and DAC combination is replaced by a single quantizer which takes the many-bit loop-filter output and generates a lower-bit word. Since everything is operating in the digital domain no DAC is required and the m-bit word can directly be used as feedback value. The noise-shaped m-bit signal is the final Sigma-Delta output. This m-bit signal is often passed to a DA converter, resulting in a Sigma-Delta DAC. In the case of audio encoding for Super Audio CD the 1-bit output is the final goal of the processing and is directly recorded on disc.

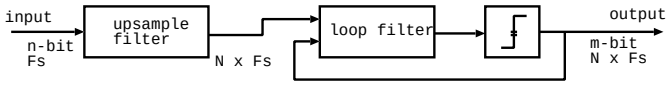


Figure 2.5: Main building blocks of a Sigma-Delta digital-to-digital converter.

2.1.3 DA conversion

A Sigma-Delta based DA converter realizes a high SNR with the use of a DAC with few quantization levels and noise-shaping techniques. In the digital domain the input signal to the DAC is shaped, such that the quantization noise of the DAC is moved to high frequencies. In the analog domain a passive low-pass filter removes the quantization noise, resulting in a clean baseband signal. The structure of a Sigma-Delta DAC is, except for some special PWM systems, a feed-forward solution,

i.e. there is no feedback from the analog output into the noise-shaping filter. Because the noise-shaping feedback signal is not crossing the analog-digital boundary, the name Sigma-Delta DAC is confusing and misleading. A Sigma-Delta DAC is the combination of a DD converter and a high-speed few-bit DAC. In fig. 2.6 the complete Sigma-Delta DAC structure is shown. The digital n-bit input signal is passed to a DD converter which upsamples the input to N x Fs before an all digital SDM reduces the word-length. The noise-shaped m-bit signal is passed to the m-bit DAC which converts the digital signal to the analog domain. Finally the analog signal is filtered to remove the out-of-band quantization noise.

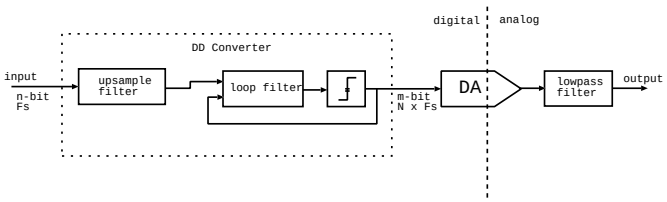


Figure 2.6: Main building blocks of a Sigma-Delta digital-to-analog converter.

2.2 Sigma-Delta structures

In sec. 2.1 it was shown that two basic SDM types exist, i.e. with an analog or a digital loop filter. In the case of an analog filter the combination of a quantizing ADC and a DAC is required for closing the noise-shaping loop and a decimation filter is present at the output. In the case of a digital filter no analog-digital domain boundary has to be crossed and only a digital quantizer is required, but at the input an upsample filter is present. When studying the noise-shaping properties of an SDM from a high-level perspective these analog-digital differences can be safely ignored and a generic model of the Sigma-Delta noise-shaping loop can be used instead. This generic model, consisting of a loop filter and a quantizer, is depicted in fig. 2.7. The loop filter has two inputs, one for the input signal and one for the quantizer feedback signal, where the transfer function for the two inputs can be complete independent in theory. In practice large parts of the loop-filter hardware will be shared between the two inputs. A practical loop-filter realization will consist of addition points, integrator sections, feed-forward coefficients b_i and feed-

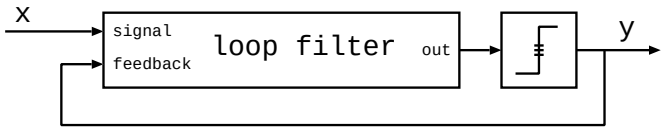


Figure 2.7: Generic model of the Sigma-Delta noise-shaping loop, consisting of 2-input loop filter and quantizer.

back coefficients a_i as shown in fig. 2.8. In this structure the number of

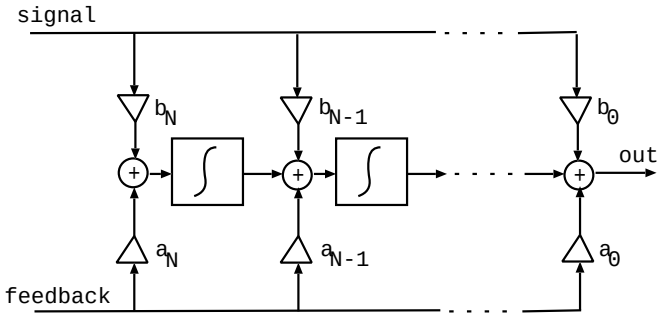


Figure 2.8: Internal structure of practical 2-input loop filter, consisting of integrators, subtraction points, feed-forward coefficients b_i and feedback coefficients a_i.

integrator sections sets the filter order, e.g. 5 concatenated integrators results in a fifth order filter. The exact filter transfer is realized by the coefficients. With proper choice of b_i and a_i the complexity of the filter structure can be reduced, e.g. resulting in a feed-forward structure. This optimized structure can be redrawn to give a 1-input loop filter where the first subtraction is shifted outside the filter, as depicted in fig. 2.9. As an alternative it is possible make all b_i equal to zero except for b_N and realize the noise-shaping transfer using only a_i. This structure is referred to as a feed-back SDM and is shown in fig. 2.10. The two structures can be made to behave identical in terms of noise shaping but will realize a different signal transfer. In both structures the quantizer can have any number of quantization levels. In practice values between 1-bit (2 levels) and 5-bit (32 levels) are used.

As an alternative to the single-loop SDM with multi-bit quantizer, a cascade of first-order Sigma-Delta Modulators can be used. This structure

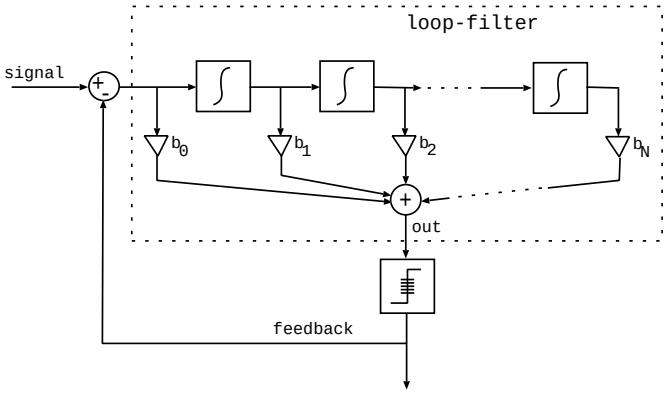


Figure 2.9: SDM with feed-forward loop filter. The subtraction point of signal and feedback has been shifted outside the loop filter.

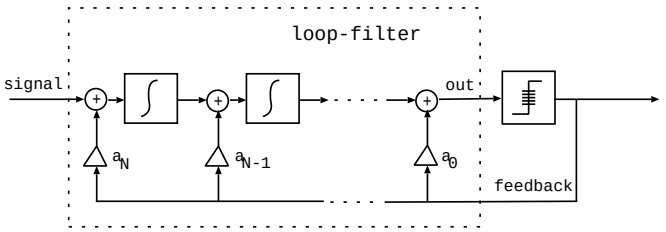


Figure 2.10: SDM with feed-back loop filter.

is commonly referred to as multi-stage noise shaping (MASH) structure. In a MASH structure the quantization error of a first modulator is converted by a second converter, as depicted in fig. 2.11. By proper weighing the two results in the digital domain with filters $H1$ and $H2$ the quantization noise of the first modulator is exactly canceled and only the shaped noise of the second modulator remains. In this fashion an n^{th} order noise shaping result can be obtained by using only first order converters. The disadvantage compared to a single-loop SDM is the inability to produce a 1-bit output.

Closely related to the SDM is the noise shaper structure. In a noise shaper no filter is present in the signal path and only the quantization error is shaped. This is realized by inserting a filter in the feed-back path which operates on the difference between the quantizer input and

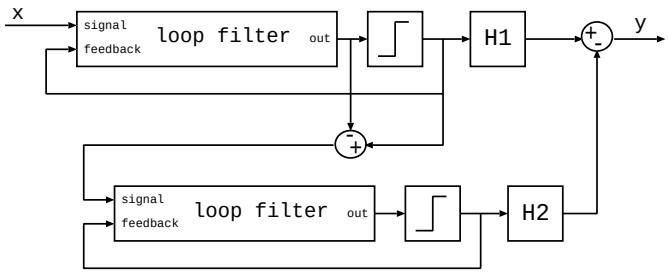


Figure 2.11: Second order MASH SDM.

quantizer output, as depicted in fig. 2.12. With a proper choice of the filter the same noise shaping can be realized as with an SDM. Unique for the noise shaper is that only the error signal is shaped and that the input signal is not filtered. Because of this special property the noise shaper can also be used on non-oversampled signals to perform in-band noise shaping. This technique is, for example, used to perform perceptually shaped word-length reduction for audio signals, where 20-bit pulse-code modulated (PCM) signals are reduced to 16-bit signals with a higher SNR in the most critical frequency bands at the cost of an increase of noise in other frequency regions.

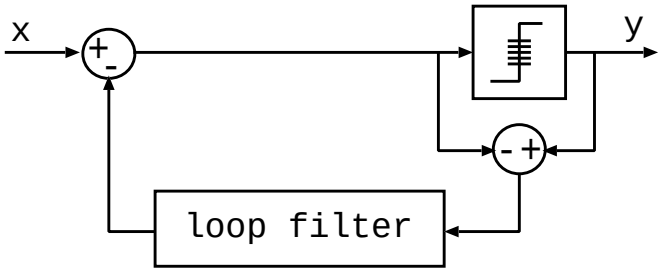


Figure 2.12: Noise shaper structure.

2.3 Linear modeling of an SDM

For a generic discrete-time SDM in feed-forward configuration, as depicted in fig. 2.13, the signal transfer function (STF) and noise transfer

function (NTF) will be derived on the basis of a linear model. In this figure $x(k)$ represents the discrete-time input signal, $d(k)$ the difference between the input and the feedback signal (the instantaneous error signal), $H(z)$ is the loop filter, $w(k)$ the output of the loop filter (the frequency weighted error signal), and $y(k)$ is the output signal.

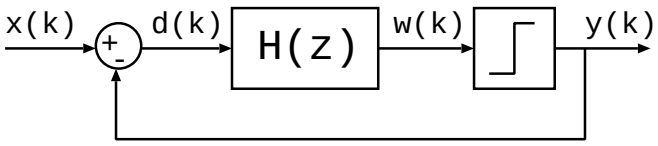


Figure 2.13: Generic model of a digital SDM in feed-forward configuration.

The difference between the quantizer output $y(k)$ and quantizer input $w(k)$ is the quantization error $e(k)$. For the schematic we can write:

$$\begin{aligned} y(k) &= w(k) + e(k) \\ &= H(z) \cdot [x(k) - y(k)] + e(k) \end{aligned} \tag{2.1}$$

$$y(k) \cdot [1 + H(z)] = H(z) \cdot x(k) + e(k) \tag{2.2}$$

$$y(k) = \frac{H(z)}{1 + H(z)} \cdot x(k) + \frac{1}{1 + H(z)} \cdot e(k) \tag{2.3}$$

From eq. 2.3 it can be seen that the output signal $y(k)$ consists of the sum of a filtered version of the input $x(k)$ and a filtered version of the quantization error $e(k)$.

If it is assumed that the quantization error is not correlated with the input signal, the quantizer can be modeled as a linear gain g and an additive independent noise source $n(k)$ which adds quantization noise. The resulting linear SDM model is depicted in fig. 2.14.

By replacing $e(k)$ in eq 2.3 with $n(k)$ and moving gain g into filter $H(z)$, the output $y(k)$ can now be described as

$$y(k) = \frac{H(z)}{1 + H(z)} \cdot x(k) + \frac{1}{1 + H(z)} \cdot n(k) \tag{2.4}$$

By setting $n(k) = 0$ the signal transfer function (STF) is obtained:

$$STF_{\mathrm{FF}}(z) = \frac{y(k)}{x(k)} = \frac{H(z)}{1 + H(z)} \tag{2.5}$$

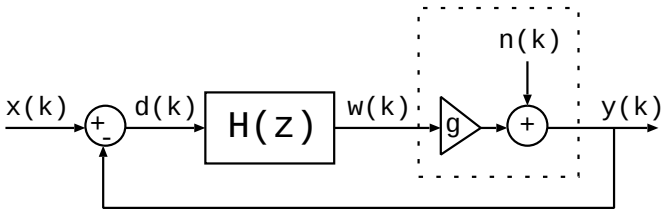


Figure 2.14: Linear model of a digital SDM in feed-forward configuration.

The signal transfer function is specific for the feed-forward structure, indicated by the subscript FF.

The noise transfer function (NTF) describes how the quantization noise, which is introduced by the quantization operation, is transferred to the output of the modulator. It is obtained by setting $x(k) = 0$ in eq. 2.4:

$$NTF(z) = \frac{y(k)}{n(k)} = \frac{1}{1 + H(z)} \tag{2.6}$$

In order to realize a high signal to noise ratio in the baseband, the quantization noise should be suppressed for low frequencies and shifted to high frequencies. As a result the loop filter $H(z)$ should be a filter that provides a lot of gain for low frequencies and little gain for high frequencies, i.e. a low-pass characteristic. With $H(z)$ low-pass it can be appreciated that the STF will be close to unity for low-frequencies and that the input signal will be accurately captured. The transfer characteristic of a typical 5th order loop filter is plotted in fig. 2.15. In this example the loop filter is designed according to a Butterworth specification for a corner frequency of 100 kHz when the sampling rate is 2.8 MHz (a 64 times oversampled 44 100 Hz system). Resonators (linear feedback within the loop filter) at 12 and 20 kHz have been added for increasing the SNR [8, 59].

With $H(z)$ given, the linearized STF and NTF can be plotted using eq. 2.5 and eq. 2.6. The result for the STF for a feed-forward (FF) as well as a feed-back (FB) modulator is plotted in fig. 2.16 for an assumed quantizer gain of 1.0. As expected, the STF equals unity for low frequencies for both types. Around the corner frequency of the feed-forward filter a gain of approximately 7 dB is realized before the filter starts to attenuate the input signal. At $Fs/2$ the input is attenuated by about 7 dB. The feed-back filter realizes a gain of approximately 3 dB at the corner frequency and then falls off strongly.

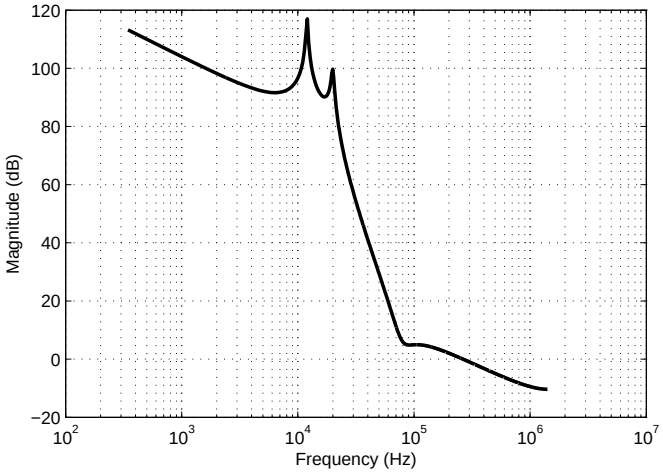


Figure 2.15: Transfer of a typical 5th order loop filter designed according to a Butterworth specification with 100 kHz corner frequency and additional resonator sections at 12 and 20 kHz. The sampling rate is 2.8 MHz.

Plotting the NTF accurately is far less trivial. It has to be realized that eq. 2.6 will only give a rudimentary approximation of the actual quantization noise spectrum, i.e. in eq. 2.6 the quantization noise is treated as an independent signal whereas in reality the signal is depending on the quantizer input. Only if signal $e(k)$ is uncorrelated with the input signal, eq. 2.6 will accurately describe the quantization noise. In the case of a multi-bit quantizer the quantization error is reasonably white for typical input signals. If desired, it can be made completely white by adding to the quantizer input a dither signal with triangular probability density (TPDF) that spans two quantization levels [63]. In the case of a single-bit quantizer the quantization error is strongly correlated with the input signal. Furthermore, since only two quantization levels exist it is not possible to add a TPDF dither signal of large enough amplitude to the quantizer input without overloading the modulator. In the case of a single-bit quantizer a deviation from the predicted NTF is therefore to be expected. Typical effects caused by the gross non-linearity of the 1-bit quantizer are signal distortion, idle tones, and signal dependent baseband quantization noise (noise modulation).

In fig. 2.17 the linearized NTF resulting from the 100 kHz filter is plotted for an assumed quantizer gain of 1.0. According to this prediction

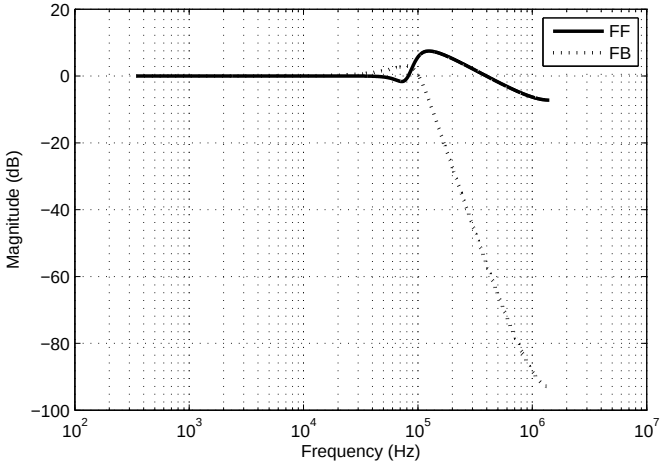


Figure 2.16: Linearized signal transfer function for the 5th order loop filter of fig. 2.15 in feed-forward and feed-back configuration (quantizer gain of 1.0).

the quantization noise will be rising with 100 dB per decade and from 100 kHz onwards the spectrum will be completely flat. At 12 and 20 kHz a notch in the quantization noise floor should be present. By means of simulations the accuracy of this prediction will be verified. For a modulator with 1-bit quantizer the output spectrum for a 1 kHz input sine wave with an amplitude of -6 dB is plotted in fig. 2.18 in combination with the predicted quantization noise spectrum. The FFT length used is 256k samples. The spectrum has been power averaged 16 times in order to obtain a smooth curve (see app. A). In the figure the predicted 100 dB per decade rise of the noise can be clearly identified. The high frequency part of the spectrum, however, deviates strongly from the prediction, i.e. a tilted noise floor with strong peaking close to $Fs/2$ is identified. In the baseband part of the output odd signal harmonics can be identified, which are not predicted by the linear models STF. The predicted notches at 12 and 20 kHz are present. As a second example, for the same modulator the output spectrum for a DC input of 1/128 is plotted in fig. 2.19 and compared with the predicted quantization noise spectrum. The spectrum shows globally the same noise shaping as in the first example, with superimposed on it a large collection of discrete tones. These so called idle tones can not be understood from the linear model, but can clearly be an issue as they are not only present at high

frequencies but also in the baseband.

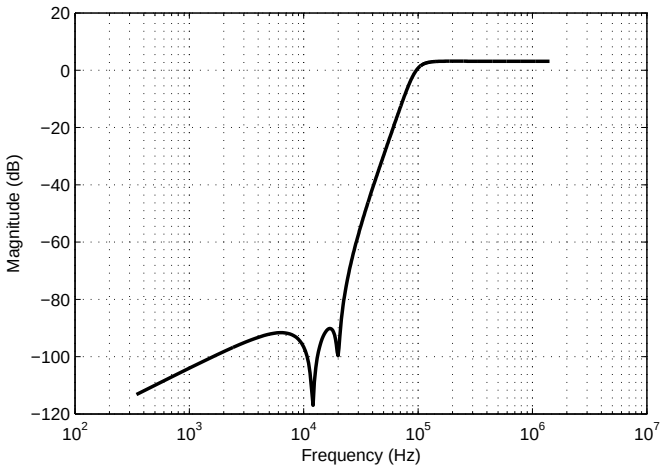


Figure 2.17: Linearized noise transfer function of the 5th order loop filter of fig. 2.15 (quantizer gain of 1.0).

As is clear from the two examples, large differences can exist between the prediction based on the linear model and actual modulator output in the case of a 1-bit quantizer. Since no accurate mathematical models for predicting a modulators response exist, the only reliable solution for obtaining performance figures of a 1-bit SDM is to perform time-domain simulations and analyze these results. Unfortunately, at the start of a design no realization exists yet and the linearized STF and NTF formulas have to be used for designing the initial loop filter. As a next step, computer simulations will have to be used to verify the response. Depending on the simulation outcome parameters will be iteratively adjusted until the desired result is obtained. In order to obtain reproducible and comparable results, in this thesis the iterative approach for designing loop filters is not taken. Filters are designed using the linear model of a traditional SDM, according to a predetermined criterion, and used as-is. The predetermined criterion will typically be a transfer characteristic according to a Butterworth prototype filter with a specified corner frequency. The actual resulting transfer might be varying as a function of the input signal and the noise-shaping structure used, and can therefore only be compared by keeping the same filter which is designed using one and the same standard approach.

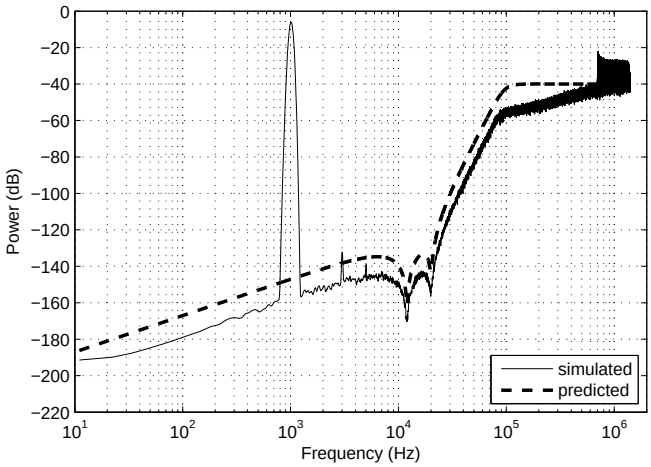


Figure 2.18: Simulated output spectrum of an SDM with 1-bit quantizer and loop filter of fig. 2.15. Input signal is a 1 kHz sine wave with an amplitude of -6 dB. The predicted quantization noise spectrum is indicated as a dashed line.

2.4 Sigma-Delta Modulator performance indicators

The performance of an SDM can be expressed in terms that describe the quality of the signal conversion process, as well as in terms of resources or implementation costs. The signal conversion performance can again be divided in two groups, namely performance measures that hold for data converters in general, and Sigma-Delta converter specific functional performance. The SDM specific functional performance indicators, discussed in sec. 2.4.2, relate to the stability of the converter, limit cycle and idle tone behavior, noise modulation, and transient performance. In order to enable an easy comparison of designs, often a Figure-of-Merit (FoM) calculation is used. In the FoM several performance indicators are combined into a single number that represents the efficiency of a design. In the case of an SDM this approach is not straightforward, and the topic is therefore discussed separately.

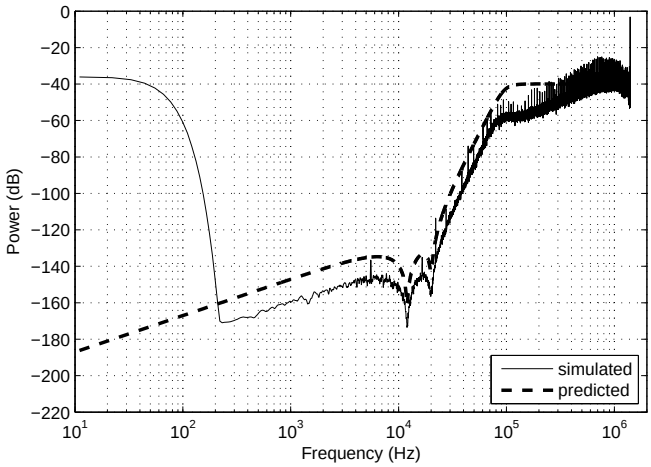


Figure 2.19: Simulated output spectrum of an SDM with 1-bit quantizer and loop filter of fig. 2.15. Input signal is a DC of 1/128. The predicted quantization noise spectrum is indicated as a dashed line.

2.4.1 Generic converter performance

The most often used generic data converter performance indicators are the Signal-to-Noise-Ratio (SNR), the Signal-to-Noise-and-Distortion-Ration (SINAD or SNDR), the Spurious-Free-Dynamic-Range (SFDR), and Total-Harmonic-Distortion (THD). Next to these signal conversion performance metrics, the implementation and resource costs are important quality aspects of a converter. By combining several of these performance indicators into a FoM, the converter performance can be specified with a single value.

SNR and SINAD

The SNR and SINAD are two closely related measures. In both cases the harmonic signal power is compared to the power of the residual (noise) signal. The residual power can be split in noise and signal distortion components. In a SINAD measurement no differentiation between the two types of signal is made and the complete residual signal is integrated, hence the name Signal-to-Noise-and-Distortion ratio. In an ideal SNR measurement only the noise part of the residual signal is integrated.

In practice however, an SNR measurement will typically only ignore the harmonically related signal components. Non-harmonically related components, i.e. combinations of the input signal frequency and the clock frequency, are often treated as noise. The SNR figure is typically slightly higher than the SINAD value because of the absence of the harmonic components. Only in the case of no distortion the two numbers are equal.

In case of a Nyquist converter the noise integration is typically performed over the complete frequency band from 0 to $Fs/2$. In the case of an oversampled converter, e.g. an SDM, the integration is performed over the band of interest only. In this thesis the band of interest is the baseband part of the output, i.e. the frequency span of 0 to 20 kHz.

Since only part of the output spectrum is used for the SINAD calculations, the SINAD will typically show a strong input frequency dependency. Typical distortion of an SDM consists of odd harmonic components, i.e. components at $(2n + 1) \cdot f_{in}$. As an example, if the input frequency is chosen as 5 kHz, there will be harmonic components at 15 kHz, 25 kHz, 35 kHz, etc. Since only the baseband (0-20 kHz in most examples) is considered for SINAD calculations, only the component at 15 kHz will be taken into account. The SINAD value for this input frequency will therefore be most likely higher than for a slightly lower input frequency which has multiple harmonics in the baseband. In order to get a single representative SINAD number, i.e. one which takes most harmonic distortion components into account, in most experiments an input frequency of 1 kHz is used. For this frequency the first 19 harmonics fall within the signal band.

In fig. 2.20 an example SDM output spectrum is shown. The input signal which has an amplitude of 0.5 (-6.02 dB) is visible at approximately 1 kHz (992 Hz), and harmonics at 3 kHz and 5 kHz are also clearly visible. In this example the SNR equals 113.2 dB and the SINAD equals 111.5 dB. The difference of 1.7 dB is primarily caused by the power in the third harmonic (HD3) and the fifth harmonic (HD5). Note that it is in general not possible to accurately read the SNR or SINAD value directly from a spectral plot - integration over all frequency bins is required and the spectral density per bin is a function of the number of points of the FFT. If a large distortion component is present in the output a rough estimate of the SINAD can be made by subtracting the power of this component from the power of the fundamental.

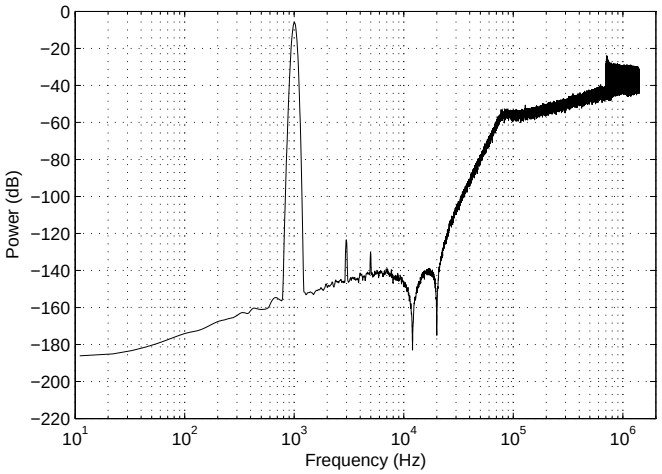


Figure 2.20: Example SDM output spectrum. FFT length is 256000 samples, 16 power averages have been performed.

SFDR

The SFDR is the difference in power between the test signal and the largest non-signal peak in the spectrum. The non-signal peak can be harmonically related but this is not required. In oversampled systems not the complete spectrum is taken into account, only the band of interest is considered. In the case of a digital SDM no artifacts other than those generated by the modulator itself are expected to be present, therefore typically the biggest peak is a harmonic component or the in-band rising noise-floor. In the example spectrum of fig. 2.20 the third harmonic is the biggest non-signal component with a power of -123.4 dB, resulting in an SFDR of -6.0 dB - -123.4 dB = 117.2 dB.

THD

The THD is the ratio between the power in all the harmonic components and the signal power. In oversampled systems only the harmonic power in the band of interest is included in the calculation. The THD value relates to the linearity of a converter, i.e. a lower THD value means less signal dependent distortion. The THD is often a function of the input level. In analog converters large inputs typically cause circuits to

saturate or clip and therefore generate distortion. In a digital SDM saturation and clipping can be avoided by using large enough word widths, but a 1-bit SDM will still generate harmonics, especially for large input signals. Determining the THD accurately can be difficult when the harmonic distortion components are of the same order of magnitude as the random noise components. In order to still get accurate results the technique of coherent averaging can be applied. The result of this process is that random frequency components are suppressed while coherent (signal) components are not. Every doubling of the number of averages reduces the random signals by 3 dB, e.g. performing 32 averages reduces the noise floor by 15 dB. Please refer to app. A for more details.

In the example of fig 2.20 the THD equals -116.3 dB, i.e. the combined power of all the harmonic components is 116.3 dB less than the power in the 1 kHz signal tone.

Implementation and resource costs

The costs of making a data converter fall in three main categories. First, there is the time required to design the converter. Second, there is the cost associated with the physical IC realization, i.e. materials and processing cost. Third, there is the cost related to the industrial testing of the manufactured device. Next to these cost factors which are occurring only once, there is a reoccurring cost factor, i.e. the cost associated with the use of the converter. This cost manifests itself as the power consumption of the converter.

Both the silicon area and required design time depend on the type and the specifications of the converter, as well as on the experience of the designer. In general it holds that if the performance specification is more difficult to reach, the required design time will be longer and often the circuit will be bigger. The power consumption of the circuit typically also scales with the area and the performance level. For example, in AD converters often thermal noise is limiting the SNR. In order to increase the SNR, i.e. reduce the thermal noise, typically a larger current is required, which in turn requires larger active devices. A data converter that uses little power is preferred over a converter that requires a lot of power. A smaller silicon area results in less direct manufacturing cost. However, the industrial testing that is required can add significant cost. A converter that requires little testing is therefore preferred over a converter that requires a lot of tests.

Figure-of-Merit

Comparison of the power efficiency of two AD converters that achieve identical signal conversion specifications, i.e. have the same sampling rate and realize the same SNR for every input signal, is an easy task; the one with the lowest power consumption is the best. However, if the signal conversion specifications are not 100% identical, the comparison becomes difficult. To overcome this problem and make the comparison of different data converters possible, typically a Figure-of-Merit (FoM) is calculated. In the FoM a single value is used to represent the performance specifications of the converter, typically the power consumption and the signal conversion bandwidth and resolution.

Unfortunately, no universally agreed standard exists for calculation of the FoM. An often used FoM equation for the characterization of AD converters equals

$$FoM = \frac{P}{2^{ENOB} \cdot min(\frac{Fs}{2}, ERBW)} \tag{2.7}$$

In this equation P equals power, $ENOB$ equals the number of effective bits measured for a DC input signal, Fs equals the sampling rate, and $ERBW$ is the effective resolution bandwidth. The $ENOB$ is calculated as

$$ENOB = \frac{SINAD - 1.76}{6.02} \tag{2.8}$$

where the SINAD is measured for a (near) DC input. The effective resolution bandwidth is equal to the frequency that results in a 3 dB SINAD reduction compared to the SINAD at DC. The unit of the FoM of eq. 2.7 is Joules per conversion step. As a result, a lower value is better. Sometimes the inverse of eq. 2.7 is used such that a higher FoM number represents a better result.

Although the FoM of eq. 2.7 is widely used, it cannot be used to make fair comparisons between low resolution and high resolution AD converters. When the resolution of an ADC is increased, a point is reached where thermal noise is limiting the SNR. In order to reduce the impact of the noise by 3 dB, capacitances need to be doubled. To increase the number of effective bits by one, a 6 dB reduction of the noise is required, which means a factor four increase in capacitance. Since power scales linearly with the amount of capacitance to charge, the power will also increase with a factor four. Thus, the FoM will become at least a factor 2 worse

when the $ENOB$ is increased by one. To enable the comparison of different resolution AD converters, an alternative version of the FoM is therefore sometimes used:

$$FoM = \frac{P}{2^{2 \cdot ENOB} \cdot min(\frac{Fs}{2}, ERBW)} \tag{2.9}$$

The equation is identical to eq. 2.7, except that the denominator becomes four times larger instead of two times when the ENOB is increased by one.

Whereas comparison of AD converters by means of a single FoM is common practice, for DA converters it is not a standard approach. One of the main reasons why for DACs the single FoM approach is problematic is the time continuous output signal. When the DAC output signal is switching, i.e. making a transition between two levels, it can follow any trajectory before the signal settles to the correct value. Deviations from the ideal switching trajectory will add noise and distortion to the output. Depending on the type of application, these glitches could be problematic but not necessarily. In some applications only the DC transfer is important whereas in other applications the signal quality over a large bandwidth is important. Sometimes a signal overshoot at a transition is allowed, sometimes a smooth settling curve without overshoot is required. However, avoiding time domain glitches will typically cost power, and therefore the power efficiency of a converter can vary greatly depending on the time domain behavior.

Another reason why the single FoM approach is difficult to apply to DACs, is that part of the power consumption of a DAC is useful, and not overhead as in the case of an ADC. The output signal of a DAC is not only an information signal, but at the same time a power signal. Typically the DAC output drives a 50Ω or 75Ω load. If a larger output swing is required from the DAC, more power will have to be spent in the generation of this signal. A higher power consumption is thus not necessary equal to less performance, but could also indicate more performance.

In conclusion, for comparing DAC performance sometimes the FoM of eq. 2.7 is used, but no actual de facto standard exists. However, since part of the power consumption is, by definition, required to drive the load, straightforward application of eq. 2.7 can lead to incorrect conclusions. Other FoM measures used for DAC characterization include the SFDR, THD, and SNR, but also the static differential non-linearity (DNL) and the integrated non-linearity (INL), as well as time domain

glitch energy measures.

2.4.2 SDM specific functional performance

The SDM specific functional performance indicators relate to the stability of the converter, limit cycle and idle tone behavior, noise modulation, and transient performance.

Stability

Higher order Sigma-Delta Modulators are conditionally stable. As a result, only signals below a certain maximum input level can be converted without causing the modulator to become unstable. This level for which the modulator becomes unstable is a function of the loop-filter order and loop-filter cutoff frequency [58]. If the loop filter is fixed, the maximum input amplitude can be determined by means of simulations.

The procedure consists of repeatedly applying a signal with a constant amplitude to the converter. The converter is run until instability is detected or until a maximum amount of time has passed. If no instability is detected within the predetermined amount of time, it is concluded that the converter is stable for the applied signal level and the signal amplitude can be increased. If instability was detected the maximum level that can be applied has been found. Instead of trying to detect instability while the converter is running, it is also possible to always run the converter for the maximum amount of time, and afterwards determine if the converter is still stable.

With the second approach it is easier to quantify the result and this is therefore the approach taken in this work. Instability can be detected by testing the output bitstream for long sequences of 1's or 0's (hundreds of equal bits), or by testing if the modulators internal integrator values are above a certain, empirically determined, threshold. The easiest procedure is to test the output bitstream. Alternatively, the SNR and the frequency of the output signal can be measured and (near) instability can be detected by testing if the obtained values differ strongly from the expected values. This is the approach taken in this work.

Instead of measuring the maximum input signal that can be handled by the modulator, it is possible to measure how aggressive the loop filter can be made before instability occurs for a given input signal. A practical test signal is a sine wave with the maximum desired amplitude. The same procedure for detecting instability as explained above can be

used, i.e. the modulator is ran for a fixed amount of time and afterwards it is determined if instability was reached. Aggressiveness of a loop filter can be increased by increasing the order of the filter or by increasing the corner frequency of the filter. Changing the filter order has a very large impact on the stability of the modulator and is therefore not practical. The loop-filter corner frequency on the other hand can be adjusted in very fine steps and is therefore more appropriate for determining stability.

In the case of a traditional SDM the stability can be determined for a given configuration, but can not be changed or influenced in any way. For the look-ahead modulator structures in this thesis the situation is slightly different, and as a function of the available computational resources the stability will vary. It is considered beneficial to have a stable modulator to enable a large input range and high SNR.

Limit cycles and idle tones

Because of the non-linear behavior of a few-bit SDM, the output signal can sometimes contain correlated frequency components that are not present in the input signal and that are not part of the normal quantization noise floor. We distinguish those components between limit cycles and idle tones. A limit cycle is a sequence of P output symbols, which repeats itself indefinitely. As a result the output spectrum contains only a finite number of frequency components. An idle tone is a discrete peak in the frequency spectrum of the output of an SDM, which is superposed on a background of shaped quantization noise. Hence, there is no unique series of P symbols which repeats itself [57]. The two situations are illustrated in fig. 2.21.

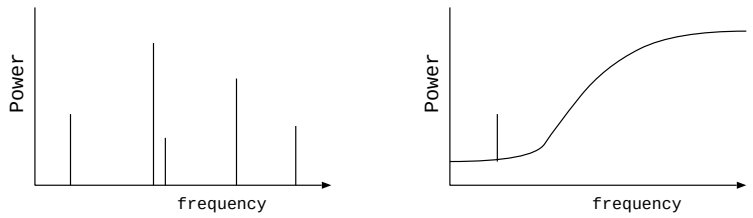


Figure 2.21: Illustration of the definition of a limit cycle (left) and an idle tone (right).

When limit cycles are present in the output signal of an SDM, typically

no signal content except DC is present at the input, although in theory also a generic repetitive input signal, e.g. a sinusoid, could be present. In practice, limit cycles only show up when the input signal is removed and a small DC offset remains. Depending on the DC level, which determines the frequency content of the limit cycle, the limit cycle can contain in-band components and cause problems or only contain harmless high frequency components.

Idle tones on the other hand typically occur when an input signal is present at the input of an SDM. Harmless high frequency idle tones are often present in the output spectrum of an SDM, but depending on the input signal the frequency of an idle tone can also be in-band and cause significant degradation of the output signal quality. Higher order modulators typically show less idle tones than low order modulators. By dithering a modulator mildly the power in idle tones can be reduced, but to fully avoid all possible idle tones a very significant amount of dither is required, penalizing the stable input range of the modulator severely. Therefore, a modulator that does not introduce idle tones or limit cycles is preferred.

Noise modulation

Noise modulation is the effect where the amount of quantization noise in the output varies as a function of the input signal power. This effect is fundamentally present for 1-bit converters. The total output power of a 1-bit SDM is constant and equals 1.0, independent of the input signal power. Since the output signal power equals the input signal power, a varying amount of the output power is available for quantization noise, and it is clear that noise modulation is required to have a functional system. Therefore noise modulation in its basic form is not an issue according to the author. The problem however is located in the fact that the amount of baseband quantization noise may vary with the input signal power.

In the case where the converter is used in an audio application, and the background noise (the quantization noise) grows stronger and weaker with changes in the music level, the effect has proven to be audible in critical listening situations. According to [14, 41, 61] the variation in background quantization noise should be less than 1 dB in order to be inaudible. For high quality audio applications the objective is to have a constant background noise which results in predictable signal quality. Therefore noise modulation should be minimized or avoided if possible.

In the special case where the converter is used in a test or measurement

setup and the only concern is to maximize the SNR for every input (AD) or output level (DD for DA), noise modulation can be used to an advantage. Since the total output power for a 1-bit converter is constant, the noise power increases when the signal power reduces. If the increase in noise power would be evenly distributed over all frequencies, the SNR in the baseband would decrease relatively more when the input power is reduced. In practice however, the amount of baseband quantization noise reduces when the input signal becomes smaller, and therefore the SNR will be higher than expected. This SNR behavior as a function of the input amplitude is depicted in fig. 2.22. The SDM used in this experiment was a fifth order with two resonators. The measured, ideal and expected SNR curves are aligned to read the same value for a -25 dB input signal. The difference between the ideal and expected curve is approximately 0.7 dB for a -5 dB input, and negligible for inputs below -10 dB.

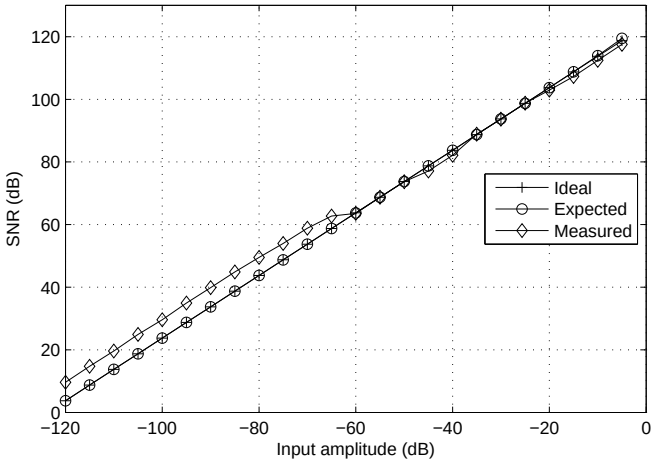


Figure 2.22: SNR as a function of the input level. Ideal behavior, expected behavior, and actual measured behavior are indicated.

The explanation of this phenomenon can be found by studying the high frequency noise spectrum. When low amplitude inputs are applied, the SDM will start to generate high frequency idle tones, which take most of the noise power. When the input amplitude is increased, the idle tones cannot exist anymore and the low frequency noise-floor will increase [57]. In fig. 2.23 the output spectrum for a -100 dB input is shown in combination with the output spectrum for a -20 dB input. The baseband

noise-floor of the -100 dB signal is significantly lower. Around $Fs/4$ strong tones are visible for the -100 dB input, whereas the -20 dB input has a small number of tones around $Fs/2$.

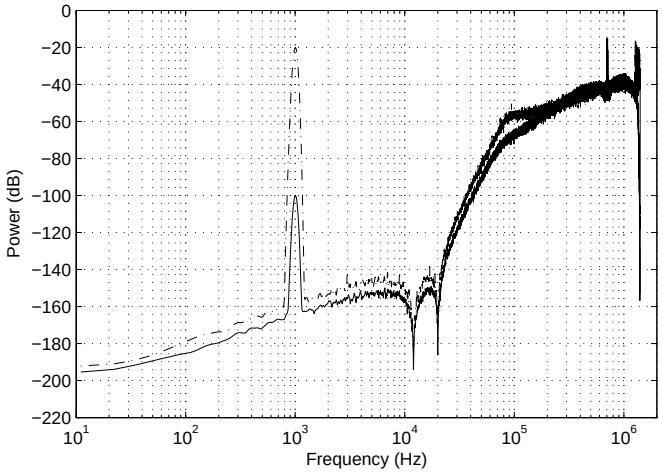


Figure 2.23: SDM output spectrum for a -100 dB and a -20 dB input. The baseband noise-floor for the -100 dB input is lower than that of the -20 dB input because more noise power is present at high frequency in the form of idle tones.

Testing for noise modulation is typically done by applying several DC levels as input signal, and for each DC level low-pass filtering the output and calculating the 2nd order moment M2 of the error. The advantage of this method is that it will measure exactly how much noise is generated for each input level. As an alternative it is possible to sweep the input level of a sine wave and to calculate the SINAD for each level. This method results in less precise results, since the sine wave passes through a range of intermediate levels, causing an average noise level. Although the amount of information obtained by performing a DC sweep is larger, the result from the AC sweep is more representative for specifying audio encoding quality. Both methods are used in this thesis.

Transient performance

Because an SDM is an oversampled system that relies on noise shaping and feedback to realize amplitude resolution, it is not under all conditions able to encode the input signal with an equally high precision.

For example, when a modulator is close to instability it can have difficulty to accurately follow transients in the input signal. When this happens, temporarily relatively large encoding errors are introduced until the modulator has recovered. Since the occurrence of this effect depends on the state of the system, it is difficult to detect or measure the impact using steady-state signals. By performing an analysis on dynamically changing signals, using a transient signal analysis method, it is possible to detect such encoding errors if they are not masked by other encoding imperfections. However, the measurement of performance in the time domain is not common practice. Therefore, a transient signal analysis method is introduced in ch. 3.

2.4.3 SDM specific implementation costs

For an SDM the implementation costs can be specified in more detail than for a generic converter, i.e. they can be associated with the specific SDM building blocks.

In the case of an SDM ADC, the converter consists of a loop filter, a quantizer, and a feedback DAC. All blocks generate noise and contribute to the final SNR. The loop filter determines the order and amount of quantization noise shaping. A higher order filter will require more components, more power, and more silicon area. Since the first integrator stage of the filter typically consumes the most power, the impact of increasing the filter order is often limited. Next, there is the quantizer. If more quantization levels are required, the complexity of this block will increase. Most commercial designs do not have more than 5 quantization bits, because the design efficiency will go down strongly for more bits. This efficiency reduction is caused by the increase of detection levels and at the same time the more stringent requirement on accuracy and noise performance of the thresholds. In most designs the number of quantization levels of the DAC equals that of the ADC. Thus, an increase in complexity of the quantizer also enforces an increase in the complexity of the DAC. Another factor that influences the cost of the building blocks is the oversampling ratio (OSR) of the converter. An increase of the OSR will require the quantizer to make its decisions faster and the DAC to produce its output faster, i.e. they run at a higher clock frequency. In most situations this will result in an increase in power consumption of those blocks. Furthermore, the output data rate of the converter also scales with the OSR, and the complexity of the subsequent digital decimation stages will also be affected. However, a higher OSR is beneficial for the noise-shaping efficiency, and might allow for a lower

order filter or less quantization levels without reducing the SNR of the converter.

In the case of a digital-to-digital SDM the implementation costs are different. All functions, i.e. the loop filter and quantizer, are realized by digital logic functions. The loop filter is by far the most difficult block and requires significant hardware resources. The filter consists of integrators, which are digitally realized by adders with feedback around a delay element, and filter coefficients. The coefficients are realized by digital multipliers or combinations of shift and add. Because typically a large dynamic range is required from the modulator, a very wide data-path is required. On the other hand, the quantizer function often involves not more than a few comparators. In the case of a 1-bit SDM, it can even be implemented without any comparison operations, and selection of the sign bit is all that is required. Also in the case of a DD converter it is the OSR of the converter that determines the clock frequency at which all the operations are performed. For most digital realizations the power consumption scales with the clock frequency. Initially this scaling is linear, but if the frequency is increased towards the limit of the technology, the scaling is more than linear. Because an SDM is a feed-back structure, pipelining of operations is not possible and all the results should be ready within a single clock cycle. The computation of the coefficient multiplication is the most challenging operation, but by selecting appropriate coefficients the computations can often be simplified.

Next to the already mentioned complexity of the loop filter, the quantizer, and possibly the feed-back DAC, another source of complexity exists for the Sigma-Delta Modulators discussed in this thesis, namely the complexity resulting from the addition of look-ahead. Without going into the details of the look-ahead concept (see ch. 5 and further), the complexity can be summarized as follows. In order to realize a look-ahead modulator, a multitude of loop filters is required. For each loop filter an alternative quantizer is present. Finally, there is a control structure that takes care of the selection of the output symbol. Because of the multiple loop filters and quantizers, the power consumption of a look-ahead modulator will be a multiple of that of a normal modulator. On top of the increased hardware complexity, there is also an increased algorithmic complexity associated with the look-ahead concept.

2.4.4 Figure-Of-Merit of an SDM

The efficiency of an SDM is typically compared using a FoM, just as is done for any other data converter. Depending on the type of SDM, i.e. an ADC, a DAC, or a DD converter, the FoM used is different.

In the case of an ADC SDM typically only the power consumption of the analog part is measured when the FoM is calculated, i.e. the digital decimation filter is often not considered in the efficiency. It is the view of the author that this is not correct, since the reconstruction filter is essential for the operation of the modulator, and can consume a considerable amount of power. Nevertheless, in practice the FoM is mostly calculated with a slightly modified version of eq. 2.7 (or eq. 2.9):

$$\frac{P}{2^{ENOB} \cdot min(BW, ERBW)} \tag{2.10}$$

In this equation BW equals the bandwidth that is used in the SNR calculation. In this equation the conversion bandwidth is thus limited to the smallest of the ERBW and the signal conversion bandwidth. Besides this change, the FoM equation is identical to the generic one and no SDM specific features are included.

For an SDM based DAC the calculation of a FoM is even more dubious than for a generic DAC. Without the digital-to-digital converter that drives the DA stage, the DAC can not work. Therefore, only by including the power of the digital SDM a sensible FoM can be calculated. The problem of selecting an appropriate FoM is now similar to the situation of a generic DAC, and eq. 2.10 is typically used.

In the case of a stand-alone DD converter, FoM calculations like the one of eq. 2.10 are typically not used. Most stand-alone converters are software based instead of dedicated hardware solutions, and therefore the power measure is not practical. A convenient metric in this case is the amount of operations per second required for the implementation to run real-time. Alternatively, the absolute amount of time can be measured that is required to process a fixed amount of signal. If the same test conditions are used repeatedly, i.e. same test signal and same computer platform, results can be compared and a valid FoM measure can be derived.

Although the above mentioned methods can be conveniently used to measure the computational efficiency of a DD design, no signal conversion performance is included in this FoM. This is not a problem if the designs under comparison are designed to deliver equal performance, i.e.

have the same loop filter and have the same OSR. If the resulting signal conversion performance is different for the converters under comparison, only measuring the computational efficiency is not good enough. However, a FoM like eq. 2.10 where P is replaced by computational load cannot be used since there is not a direct relation between the signal conversion performance and the computational load, i.e. a change of the loop-filter transfer will affect the SNR but not necessarily the amount of computations.

Because of the issues in defining a simple FoM that includes all the relevant measures, stand-alone DD Sigma-Delta Modulators will have to be compared using several measures, similar to the situation of comparing generic DACs. For designs that realize an identical SNR, the relevant signal quality measures are the SFDR and the THD. Next to these generic measures, designs can be compared on the relevant SDM specific measures, mainly the stability of the converter, the transient behavior, and possibly the amount of noise modulation. The realized signal quality can then be compared with the computational load required for this quality.

Deciding on what design is the best on the basis of the different metrics, however, is not straightforward if similar performance levels are reached. In this case the design with the lowest computational load is selected as the better one. For practical reasons, in this thesis the computational load of the different DD converters is measured by recording the time that is required to process a specific test signal. These measurements are performed on the same computer platform under the same conditions, such that the results can be used to classify performance levels. Note that the impact of the computer architecture is not considered in this performance evaluation, and that the results can not be used for generic benchmarking purposes against literature.

Chapter 3

Transient SDM performance

Signal conversion quality is typically measured using steady-state signals. However, in the case of a non-linear data converter, there is a clear motivation to also measure the performance of non-steady-state signals. An outcome of a discussion on this (sec. 3.1) is that a time-domain measurement is desired. A SINAD based measurement is proposed and introduced in sec. 3.2. An analysis of the time domain SINAD measurements for steady-state signals is made in sec. 3.3. Next, SINAD measurement results on a non-steady-state signal are discussed in sec. 3.4. Finally, the chapter is concluded in sec. 3.5.

3.1 Measuring signal conversion quality

Traditional signal quality analysis can only be performed on steady-state signals. However, since real-life signals are, typically, not steady-state and an SDM is a non-linear system, it could also be interesting to measure the conversion performance for non-steady-state signals.

3.1.1 Steady-state

Signal conversion performance characterization of a data converter is typically performed using SNR, SINAD, SFDR, and THD measurements. All these measurements are performed by means of a frequency analysis of the output signal while the input signal is a sinusoid, i.e. steady-state signal analysis.

If we consider the SINAD measurement, the analysis consists of calculating the output spectrum, locating the test tone in the spectrum and measuring the power of this tone, and calculating the ratio of the signal power to all the other power in the signal band. Thus, the ratio of the signal power to the power of the noise-and-distortion is calculated. Although the name suggests that all types of distortions are added to the noise and therefore reduce the SINAD value, this is not correct. More specifically, linear distortion operates on the signal component and can change the amplitude and/or phase of the signal, but does not introduce any additional frequency components in the output signal. Whereas a change in the signal amplitude affects the power of the output signal, a phase shift does not influence the measured output power. Non-linear distortion on the other hand, will introduce additional frequency components. The distortion can be harmonically related to the input signal, but does not need to be. Non-linear distortion, e.g. compressive behavior, can also influence the power of the output signal. Therefore, SINAD should be interpreted as the ratio of the output signal power to the power of the noise and non-linear distortion components in the output, measured for a steady-state signal. The power of the output signal is related to the power of the input signal through the linear and non-linear transfer characteristic of the converter.

3.1.2 Non-steady-state

Although real-life signals that are fed to a data converter are often not sinusoidal, the result of a performance characterization with steady-state stimuli can be considered as a representative quality indicator for the system, if the data conversion system is linear. However, in the case of a non-linear data converter the principle of superposition is not valid, and signal dependent conversion errors can be expected. Because of the noise-shaping loop of an SDM, where the loop-filter state is a function of the complete conversion history of the converter, the analysis with steady-state signals is even less representative.

In the SDM algorithm the input signal is approximated with a coarsely quantized oversampled signal. As a result, it is impossible for an SDM to instantly follow the input signal. Only the reconstructed output, i.e. a filtered or averaged version of the output signal, will follow the input. Because the oversampling ratio of a typical SDM is very high, the time-domain difference between the input signal and the reconstructed output signal is typically very small for a steady-state situation. However, since the modulator can only react on transients in the input after they have

happened, there can be momentarily small differences between the input signal and the reconstructed output signal. This difference typically depends on the internal state of the modulator before the transient, and the shape and amplitude of the transient. For example, if a modulator is close to overload it will often have difficulty following the input signal, and temporarily a relatively large transient error could result. If the amplitude of the transient is small it will be easier to approximate the signal than in the case of a large transient. The shape of the transient is also of influence on the quality of the approximation, i.e. a slow change in the input signal can be followed with less problems than a high frequency signal. Thus, it is to be expected that, especially in the case of minimally oversampled converters, the time domain errors will be typically larger for large amplitude high frequency signals than for low frequency signals with a small amplitude.

In general, the output of an SDM does not only depend on the input signal, but it is severely influenced by the conversion history and thus by the loop-filter state. Therefore, the conversion quality can vary over time, as a function of the signal dynamics. Here, conversion quality can be, for example, interpreted as the instantaneous SINAD, i.e. the SINAD measured over a relatively short time fragment. However, since the input signal is not a steady-state signal, it is not possible to perform a traditional (frequency domain) SINAD measurement. Therefore, in order to analyze the performance of an SDM for non-steady-state signals, a transient SINAD measurement method is proposed.

By comparing the results obtained from a transient SINAD measurement on a representative non-steady-state signal with the SINAD measured for a steady-state signal, it is, in the first order, possible to identify how much additional non-linear distortion and/or noise because of temporal encoding errors are added under these conditions. The outcome of this analysis is only a first order indication because an SDM is a non-linear system, and as a result the amount of non-linear distortion and quantization noise, as obtained from the steady-state analysis, could also be changing.

3.2 Time domain SINAD measurement

From the previous section it is clear that in order to measure the transient performance of an SDM, a time-domain analysis is desired, for which we have suggested a time domain SINAD measurement. The setup for the time domain SINAD measurement is depicted in fig. 3.1.

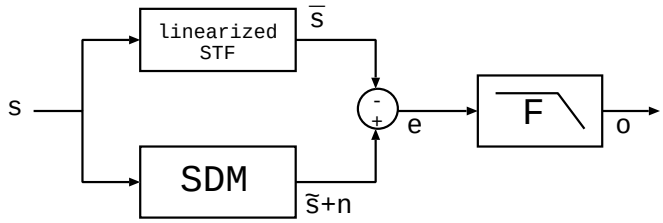


Figure 3.1: Measurement of the time domain error signal.

In the time domain SINAD approach the signal power is not directly obtained from the SDM output signal, but is calculated from a noiseless copy of the SDM output, such that there is no need to isolate the actual signal from the noise, which is difficult in the time domain. The noiseless copy, signal $\bar{s}$ in fig. 3.1, is obtained by applying the linearized STF of the SDM to the input signal s. This way the linear distortion that is introduced by the SDM to s, will be also present in $\bar{s}$. The power of $\bar{s}$ is calculated by summing the square of the samples $\bar{s}$ over the time fragment of interest. If the SINAD is calculated over the baseband instead of over the full bandwidth, which is the standard procedure for characterizing an SDM, in principle this approach is not valid, and the signal $\bar{s}$ will need to be passed through the low-pass filter F before calculating its power. However, if the low-pass filter F has a unity gain in the baseband, the signal power before and after filtering will be the same, and the filtering operation is not required.

The power of the noise-and-distortion part of the SDM output signal is calculated from the baseband part of the difference signal e, i.e. the power of a low-pass filtered version of the difference between the modulator's output and the noiseless signal copy. This is comparable to the way the noise-and-distortion is calculated in the frequency domain SINAD procedure, i.e. there the signal is removed from the spectrum by zeroing the signal bins, and the power of the bins that fall within the band-of-interest is summed. Thus, summing of the power in the baseband bins in the frequency domain method is identical to integrating the power of the time domain signal e after filtering it with a brick-wall filter, under the condition that $\bar{s}$ is identical to $\tilde{s}$. In practice, it is not possible to apply a brick-wall filter and some out-of-band noise will be included in the integration, leading to a slightly reduced SINAD. If $\bar{s}$ does not match $\tilde{s}$ exactly, e.g. because the linearized STF only approximates the actual linear distortion introduced by the modulator, not all

signal power will be removed.

With these limitations in mind, it is clear that with this procedure it is not only possible to measure the SINAD for steady-state signals, but also of non-steady-state signals.

Let the input to the SDM be the time domain signal s. Under the assumption of an ideal SDM, i.e. no signal impairments are introduced during the conversion process, the output of the SDM is an encoded version of s in combination with quantization noise, denoted by $\tilde{s} + n$. For the SINAD of the SDM output, measured over the full output bandwidth, it holds in general that

$$SINAD = \frac{\tilde{S}}{N} \neq \frac{S}{N} \tag{3.1}$$

where $\tilde{S}$ (S) represents the signal power in the SDM output (input) signal, and N represents the noise power in the output signal. Note that $\tilde{s}$ is, typically, not equal to s because of a non-unity STF, i.e. often linear distortion is introduced, resulting in an amplitude variation and a change of the signal phase.

The time domain difference between $\bar{s}$, the SDM input signal after filtering with the linearized STF, and the Sigma-Delta encoded signal $\tilde{s}$ (the signal part of the output signal) is denoted by δ:

$$\tilde{s} - \bar{s} = \delta \tag{3.2}$$

In the ideal case where the linearized STF matches the actual signal transfer of the SDM, δ will be equal to zero.

The SDM encoding error e is given by the difference between the actual SDM output signal and the noiseless replica of the SDM output signal:

$$\begin{aligned} e &= (\tilde{s} + n) - \bar{s} \\ &= \tilde{s} - \bar{s} + n \\ &= \delta + n \end{aligned} \tag{3.3}$$

If δ equals zero, the encoding error should, ideally, consist of quantization noise only. In practice, the encoding error signal will contain all the errors introduced by the Sigma-Delta encoding, e.g. idle tones and dynamic errors. However, if the in-band part of the linearized STF does not match completely with the actual signal transfer of the SDM, the encoding error signal will also contain signal related components.

The time domain SINAD value $SINAD_{\mathrm{TD}}$ over the complete output bandwidth is now calculated as

$$\begin{aligned} SINAD_{\mathrm{TD}} &= \frac{\bar{S}}{E} \\ &= \frac{\bar{S}}{\Delta + N} \end{aligned} \tag{3.4}$$

where E represents the power of e and Δ is the power of δ.

By comparing eq. 3.4 with eq. 3.1 it can be easily seen that the SINAD as obtained from the time domain signal will only be equal to the SINAD as measured using the frequency domain method if the signal $\tilde{s}$ equals $\bar{s}$ exactly, such that Δ equals zero. In-band deviations of the linearized signal transfer will reduce the time domain SINAD compared to what is calculated using the traditional frequency domain approach.

For an SDM the SINAD is normally calculated over the signal band instead of over the full output bandwidth. If an ideal low-pass filter F is assumed, i.e. a unity gain in the signal band and an infinite suppression outside the signal band, the signal power will not be affected by the filtering operation, and the out-of-band noise will be removed. Thus, by measuring the noise at the output of the low-pass filter F it is possible to calculate the SINAD over the signal band.

For calculating the time domain SINAD over the pass-band, the power of o is used instead of the power of e:

$$SINAD = \frac{\bar{S}}{O} \tag{3.5}$$

Again, the result will only be accurate if the signal $\tilde{s}$ equals $\bar{s}$ exactly, such that signal o does not contain any signal components. This means that the signal spectrum at the output of the linearized STF function is equal to the signal spectrum at the output of the SDM.

3.3 Steady-state SINAD measurement analysis

The difficulty with the time domain SINAD measurement procedure of the previous section is that the linear distortion, introduced by the STF, has to be modeled correctly by the linearized STF block in order to get a match between the time domain and the frequency domain SINAD measurement. Although from the linear SDM model a prediction of the STF can be obtained (see eq. 2.5 in sec. 2.3), in practice it is not straightforward to obtain an accurate prediction, because the linearized

STF is only an approximation of the actual STF and depends on the effective quantizer gain. The effective quantizer gain depends on the signal at the input of the quantizer, that is a function of the input signal. Thus, the 'linearized STF' should depend on the signal level and should, strictly seen, be replaced by a non-linear function. However, for a single amplitude level it is possible to obtain a reasonably accurate linearized STF, such that reliable time domain SINAD measurements are possible.

3.3.1 Obtaining the linearized STF

An accurate linearized STF will be derived by first measuring the linear distortion introduced by an SDM, and afterwards fitting the linearized STF to the obtained results. For this procedure a time domain SINAD measurement is performed with the linearized STF function replaced with a unity gain and zero delay, such that the linear distortion of the SDM is not canceled. The difference between the SINAD obtained using this procedure and the frequency domain method is resulting from the linear distortion. From this result it is possible to derive a linearized STF that resembles the actual STF of the SDM. This approach can be followed since the frequency domain SINAD measurement results (fig. 3.2) show no frequency dependencies, which indicates that no frequency dependent non-linear distortion is generated by the SDM, and as a result all the deviations between the two curves can be accounted to linear distortion.

The time domain SINAD values, without compensating the linear distortion of the modulator, are compared to the actual (frequency domain) SINAD values for three different amplitude levels for frequencies spanning the complete audio band. For the experiment a fifth order SDM with two resonators is used. The results are depicted in fig. 3.2.

For the -86 dB signal level the results obtained using the two methods agree with great precision. For the -46 dB signal level there are some small discrepancies between the two measurements. However, for the -6 dB input level large frequency dependencies are visible in the time domain measurement result. Only at the resonator frequency of 12 kHz the measured SINAD values are identical. For the other frequencies the difference between the actual (frequency domain) SINAD value and the time domain result can be as large as 20 dB. Thus, for low amplitude signals the STF can be fairly well approximated with a unity gain transfer, but for large amplitude signals this results in large discrepancies between the actual SINAD value and the value found using the time

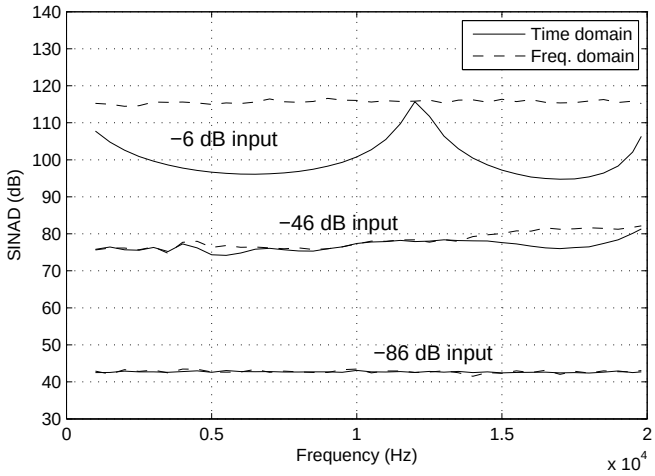


Figure 3.2: SINAD as a function of the input frequency, measured in the time domain and in the frequency domain. For the time domain measurement a pure unity gain STF is assumed. Input levels are -86 dB, -46 dB, and -6 dB. Loop filter is 5th order feed-forward, 100 kHz corner frequency, resonators at 12 and 20 kHz.

domain method. Therefore, the linearized STF will be derived on the basis of the -6 dB input signal results.

From the linearized STF of a typical SDM, as shown in fig. 2.16, it seems that for the 0-20 kHz frequency range the gain transfer equals unity and that a zero phase shift is realized. However, a zoom-in on the baseband of the STF curve (fig. 3.3) reveals that this is not the case. The small deviations from the ideal curve for both the gain and the phase cause the reduction of the SINAD values that are obtained using the time domain method when a unity STF is used for canceling the signal, as the difference between the SDM STF and the replica linearized STF is added to the noise as signal Δ in eq. 3.4. The amount of deviation from unity of the linearized STF is a function of the assumed effective quantizer gain, i.e. the higher the quantizer gain the closer to unity the linearized STF becomes. If it is assumed that the general shape of the linearized STF matches with the actual STF, it should be possible to obtain an accurate match between the linearized STF and the actual STF if the correct effective quantizer gain is used.

The proper effective quantizer gain can be found by replacing the actual

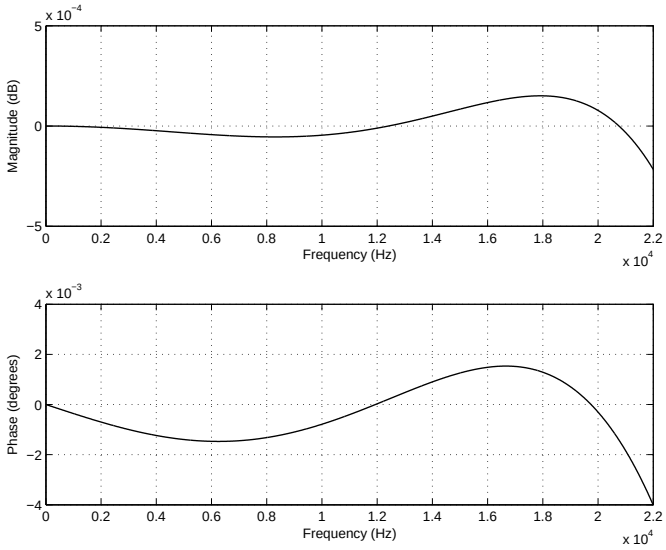


Figure 3.3: Gain and phase transfer in the signal band according to the linear SDM model with an assumed quantizer gain of 1.0.

SDM of fig. 3.1 by an SDM model, consisting of a linear STF function and a noise source to represent the quantization noise, and repeating the SINAD measurement experiment described above for different values of the effective quantizer gain. If the SINAD values obtained with the SDM model are (near) equal to the SINAD values obtained with the actual SDM the correct effective quantizer gain has been found. In the case of the -6 dB input signal a good match between the actual SINAD curve and the SINAD curve predicted on basis of the linear model can be obtained for an effective quantizer gain of 1.7.

The predicted effect on the time domain SINAD value, resulting from the gain and phase variations obtained from the linearized STF with an assumed effective quantizer gain of 1.7, are shown in fig. 3.4 and compared to the SINAD curve that is measured for the actual SDM (from fig. 3.2). If only gain errors are considered, a relatively large discrepancy is present between the prediction and the measured result. The SINAD resulting from phase errors only matches fairly well with the measurement result for low frequencies, but deviates significantly for frequencies close to 20 kHz. The predicted SINAD when both the gain and phase errors are taken into account matches within 1 dB over

the complete audio frequency range with the measurement result, and accurately predicts a SINAD reduction from 116 dB at the resonator frequency to 95 dB at 17 kHz.

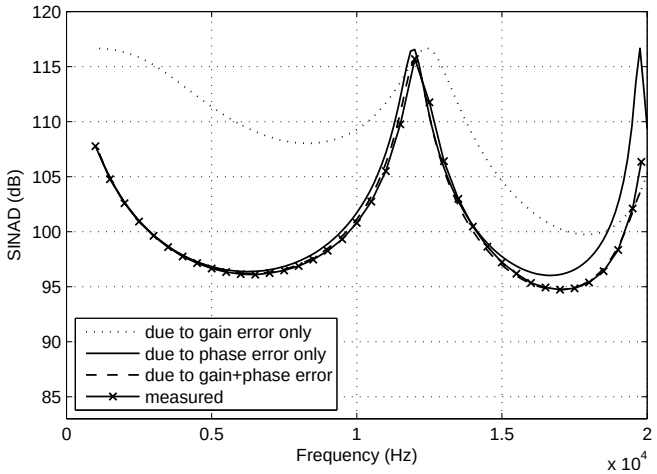


Figure 3.4: Comparison of the predicted time domain SINAD measurement result, resulting from the individual, as well as from the combined, gain and phase deviations as predicted by the linear SDM model for a fitted quantizer gain of 1.7, with the measured time domain SINAD for a zero delay unity gain linearized STF (-6 dB input signal).

Because the predicted SINAD values accurately match the results obtained by time domain simulations with the actual SDM for the -6 dB input signal, it is reasonable to assume that the frequency dependent gain and phase shift, as predicted by the linear SDM model, are indeed realized by the modulator. However, this match is only realized for the -6 dB input signal, as depicted in fig. 3.5. For the -46 dB signal small deviations are present between the SDM model and the actual SDM, and for the -86 dB signal a difference of 5 dB is recorded. These differences are caused by signal dependent variations in the effective quantizer gain and the presence of high frequency idle tones in the case of the -86 dB signal that cause a reduction of the baseband quantization noise (see sec. 2.4.2), making it difficult to reliably predict the SINAD on the basis of a single linearized SDM model for a wide range of signal amplitudes. However, for a single amplitude a reliable linearized STF can be derived, where a fitted quantizer gain accurately enough models the non-linear influence on the linear distortion, such that time domain

SINAD measurements can be performed.

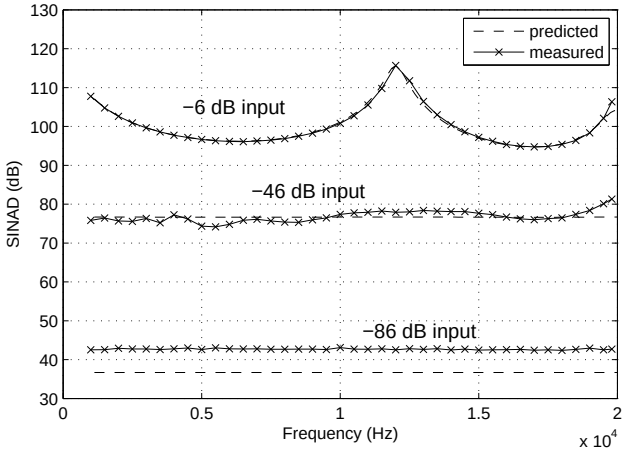


Figure 3.5: Comparison of the predicted time domain SINAD measurement result, on the basis of a fitted quantizer gain of 1.7, with the measured time domain SINAD for a -6 dB, a -46 dB, and a -86 dB input signal. The linearized STF is equal to a zero delay unity gain.

3.3.2 Time domain SINAD measurement

With the matching linearized STF available, it is now possible to perform the actual time domain SINAD measurement. The setup of fig. 3.1 is used, with the derived linearized STF in place. Again, for three levels (-86 dB, -46 dB, and -6 dB) the steady-state SINAD values for several frequencies spanning the complete audio frequency range are measured. For comparison, the measurements are also performed using the traditional frequency domain method. The results are depicted in fig. 3.6. As expected, for the -6 dB input signal both measurements are in good agreement, except for frequencies above 15 kHz where the time domain measurement reports a lower SINAD value. This can only be explained by a mismatch of the linearized STF with the actual STF. For the -46 dB and the -86 dB input signals a fairly good match is obtained over the complete frequency range and the results look similar to those obtained with a unity gain STF (fig. 3.2). Although the linear distortion is a function of the signal level, and can therefore not be modeled correctly with

a single linear STF, a fairly good result is obtained for all amplitudes with the linearized STF derived for the -6 dB input signal. Thus, reasonably reliable time domain SINAD measurements can be performed, as long as the frequency of the signal is restricted to the 1 kHz to 15 kHz range.

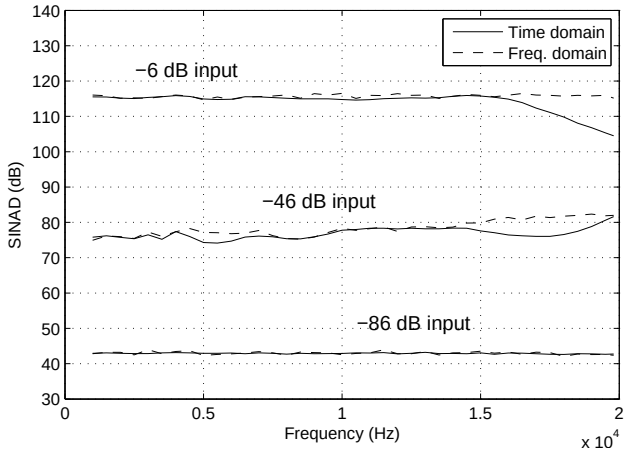


Figure 3.6: SINAD as a function of the input frequency, measured in the time domain and in the frequency domain. For all the three time domain measurements a linearized STF with an effective quantizer gain of 1.7 is assumed. The input levels are -86 dB, -46 dB, and -6 dB. The loop filter is 5th order feed-forward filter with 100 kHz corner frequency and resonators at 12 and 20 kHz.

3.4 Non-steady-state SINAD measurement analysis

The time domain SINAD measurement procedure enables the measurement of SINAD values on arbitrary signals, i.e. not only on steady-state signals. The advantage of measuring SINAD values on non steady-state signals is that also dynamic errors, if present, are included in the measurement. However, a comparison of results obtained on non-steady-state signals with results obtained on steady-state signals is not straightforward. Only when the linearized STF matches exactly with the actual STF of the SDM, the linear distortion components are excluded from the time domain SINAD measurement, and a fair comparison between the

steady-state and the non-steady-state performance can be made. However, in general it is not possible to obtain a single linearized STF that matches with great accuracy for all input frequencies and amplitudes, as was shown in previous section. Still, for the representative example SDM that was used, a good fit of the STF was realized for frequencies in the range of 1 kHz up to approximately 15 kHz, by setting the effective quantizer gain to 1.7.

As an experiment, an example non-steady-state signal has been generated by sweeping, at a high rate compared to the sampling rate of the system, the frequency of a sine wave linearly from 1 kHz to 15 kHz. By restricting the frequency content like this, under the assumption that the actual SDM STF is static, the linearized STF will match the actual SDM STF. Since the frequency of the input signal is changing fast, the input signal can be considered non-steady-state.

For this non-steady-state signal the SINAD value is calculated with the time domain SINAD method. Since the signal is non-steady-state it is not possible to calculate a reference SINAD value using the traditional frequency domain method. To enable a comparison, the SINAD value for a 1 kHz input signal is calculated and compared to the value obtained for the non-steady-state signal. No significant difference in the two SINAD values is detected. It has to be realized that if dynamic errors are present, any other signal, or possibly even the same signal with different initial conditions for the modulator, could result in a different outcome.

For the example non-steady-state signal the time domain low-pass filtered error signal o (see fig. 3.1) is visually inspected. In the top part of fig. 3.7 this time domain signal is plotted. In the bottom part of the figure the instantaneous frequency of the input signal is shown. The error signal looks like noise and no recognizable correlation with the input signal is present, which is also confirmed by a cross-correlation calculation. Thus, if dynamic errors are present, they are at a low level and can not be visually distinguished from the quantization noise.

By repeating the SINAD measurement for the example non-steady-state signal for different amplitude levels, a non-steady-state SINAD versus amplitude plot has been generated. For all amplitude levels the linearized STF has been kept constant, assuming an effective quantizer gain of 1.7. From the previous section it is known that this will introduce small errors in the obtained SINAD values, but since the errors are small this is acceptable. The results are presented in fig. 3.8. As a reference, the SINAD for a 1 kHz input signal, measured using the traditional frequency domain method, has been added to the plot. Virtually no difference can be detected between the two curves, i.e. the difference is

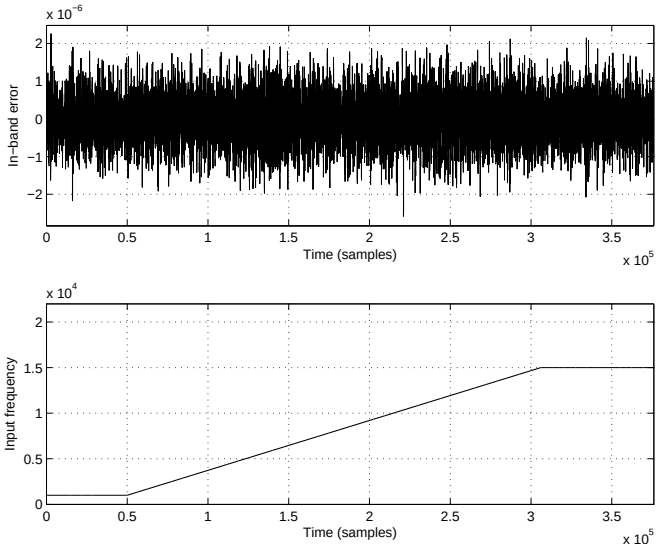


Figure 3.7: In-band SDM encoding error for a -6 dB transient signal (top) and the frequency of the input signal (bottom) over time. The sampling rate of the SDM is 2.8 MHz.

less than 1 dB for all signal levels, as expected from sec. 3.3.1. If dynamic errors are generated by the SDM when stimulated with the test signal, the level of those errors is too low to influence the SINAD values significantly. Thus, in this case, there are no errors generated because of the non-steady-state signal, or they are at a level too low to be relevant.

The same experiment has been repeated with a variety of non-steady-state signals, but none of these signals triggered the SDM to introduce errors which were significant enough to alter the SINAD values. Although this outcome is no proof for the absence of dynamic errors, it does show that under normal operating conditions the performance of an SDM is not significantly influenced by dynamic errors.

3.5 Conclusions

Traditionally, signal quality is measured using steady-state signals. This enables a frequency domain analysis, which is fast and convenient. However, real-life signals are typically not steady-state. In the case of a linear

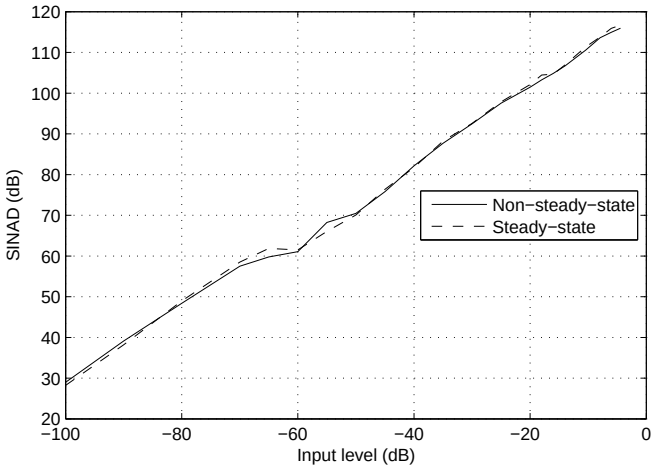


Figure 3.8: Non-steady-state SINAD (time domain measurement) and steady-state SINAD (frequency domain measurement, 1 kHz input) as a function of the input level.

data converter, the performance for these signals can be derived from the steady-state signal performance. In the case of an SDM, a non-linear data converter, such an extrapolation is not valid. To overcome this problem, a time domain SINAD measurement is proposed.

The time domain measurement procedure relies on an accurate noiseless reproduction of the SDM output signal including all the linear distortion, realized by filtering the SDM input signal with a linearized STF, such that by subtracting this signal from the actual SDM output the noise-and-distortion time domain signal can be obtained. The difficulty of this approach is that the linearized STF will have to match the actual STF with great accuracy in the signal band, or a too low SINAD value will be found. A method for deriving a linearized STF with reasonably good accuracy has been realized. It has been demonstrated that over the frequency range of 1 kHz to 15 kHz the difference between the SINAD value found using the traditional frequency domain procedure and the proposed time domain procedure is negligible. For higher frequencies the time domain method reports a lower SINAD because of a misfit of the linearized STF with the actual realized signal transfer. If the linearized STF is replaced by a pure gain, the procedure can also be used to measure the linear distortion of the converter, and it has been

shown that a signal level dependent phase/gain modulation is realized by a typical SDM.

Once the linearized STF has been derived using steady-state signals, the time domain SINAD method can also be used to measure the performance in case of non-steady-state signals. By comparing the SINAD values of steady-state signals and non-steady-state signals it can be detected if additional distortion is generated. However, none of the experiments showed a significant difference between the steady-state and non-steady-state performance. This result shows that, typically, no significant encoding errors are resulting from dynamically changing signals. However, it does not proof that under no condition the encoding quality can degrade, it only proofs that it is not straightforward to reduce the encoding quality of an SDM. Thus, although it is possible to measure the SINAD for non-steady-state signals, there is little motivation to do so, and the traditional steady-state frequency domain method will be used in the remainder of this thesis for characterizing performance.

Chapter 4

Noise-shaping quantizer model

In order to make the step from a traditional SDM, which is a noise-shaping quantizer, to a look-ahead enabled SDM an abstract noise-shaping quantizer model will be created. This model will then be used in the next chapter to derive a model of a look-ahead enabled SDM. The noise-shaping quantizer model will be derived from a generic quantizer model, which also describes an SDM from a high level of abstraction (sec. 4.1). If the model is made less abstract, it becomes clear that an SDM is not just a quantizer, but is a quantizer that performs noise shaping. In fact, it is a specific realization of the generic noise-shaping quantizer, which is again a subset of the class of generic quantizers. With these insights, in sec. 4.2 a noise-shaping quantizer model is derived from the generic quantizer model, and it is demonstrated that this model can also be applied to a normal SDM. Next, in sec. 4.3 the generic noise-shaping quantizer model is refined by adding multiple cost functions, and in sec. 4.4 two practical realization structures are introduced.

4.1 Generic quantizer

A generic quantizer is typically modeled as a block with one input and one output, as depicted in fig. 4.1. In this model the output signal is a quantized representation of the input signal. From a high-level perspective this is a fully valid approach, however, with such a simplistic model the behavior of the block is only minimally described. In practice, more details of the quantizer characteristic are required in order to be

able to use it for modeling or simulation purposes. These details, e.g. the number of quantization levels or noise-shaping characteristics, are typically described in an accompanying text and from thereon implicitly assigned to the quantizer.

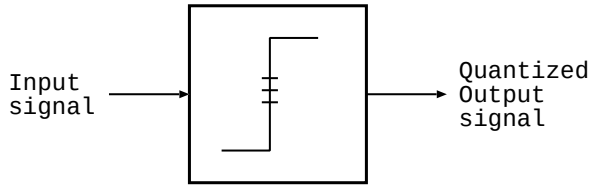


Figure 4.1: Generic two port quantizer.

However, for modeling purposes it is sometimes beneficial to explicitly include more details of the quantizer in the high-level model. For example, a second input can be added to the model that receives a set of discrete levels, as depicted in fig. 4.2. More specifically, this set of discrete levels describes what decision thresholds are available to the quantizer, and to what output code the crossing of these levels should be mapped. If the quantizer is performing analog-to-digital conversion, the threshold values will be analog levels and the output codes are digital values. In the case of a digital-to-digital conversion, both the decision thresholds and the output codes will be digital values.

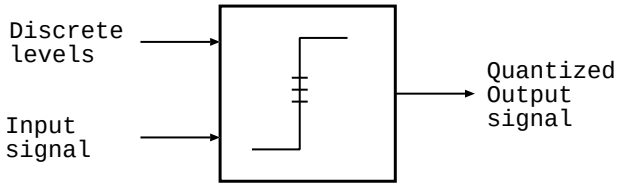


Figure 4.2: Generic quantizer model with explicit discrete levels.

Not explicitly shown in the generic quantizer model of fig. 4.2 is the presence or absence of a memory. Without a memory the quantizer can be described as a non-linear, but relatively simple, input-to-output mapping function. In this case, in theory at least, an input value will be always mapped to the same pre-determined output value without any dependency on history. In practice, the quantizer outcome might vary between a number of codes because of noise or other time-varying prac-

tical impairments like power supply variations. As long as those impairments are not a function of the signal that is converted, the quantizer is still without memory.

If the quantizer does contain an element of memory, the output code of the quantizer will not only be a function of the current input signal, but will also depend on the history of the input signal. As a result, the input-to-output mapping function will become more difficult. If the memory is realized by internal feedback around the quantizer, a quantizer with noise-shaping properties can be realized, for example an SDM. Because of the feed-back around the non-linear quantization operation, analytical modeling of such a quantizer is considered a big challenge and linear approximations of the transfer characteristics are often used (sec. 2.3).

4.2 Noise-shaping quantizer

In the previous section it was shown that a noise-shaping quantizer is a subset of the set of generic quantizer functions, and requires internal feed-back around the quantizer. This specific detail is added to the generic quantizer of fig. 4.2, resulting in the block diagram of fig. 4.3.

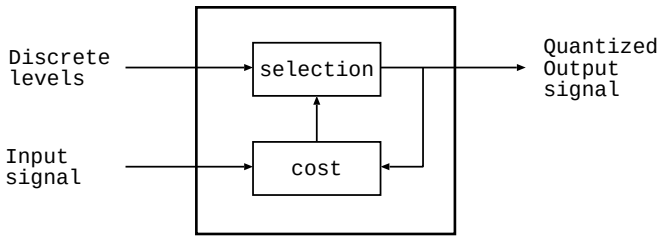


Figure 4.3: Generic noise-shaping quantizer.

In the block diagram, within the generic quantizer two blocks and a feed-back loop can be identified. The signal input is connected to a two-input cost function. The cost function receives as second signal the quantizer output code. On the basis of these two signals, and possible internal states (memory), the cost function generates an output signal which drives the selection block. The selection function subsequently maps this cost value to an output signal, selected from the set of discrete levels. The output of the selection block is supplied as the final noise-shaping quantizer output value, and also passed back to the cost function. Thus,

there is a feed-back loop around the cost function and the selection function.

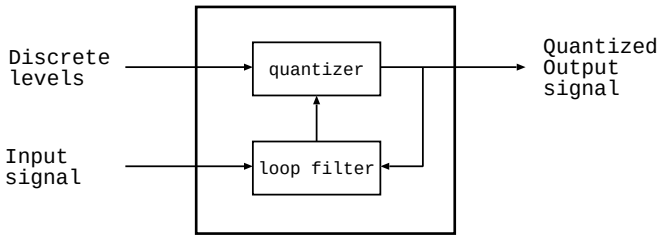


Figure 4.4: Conventional SDM implemented as generic noise-shaping quantizer.

By making the blocks in the structure of fig. 4.3 more specific, it will be shown that the schematic can represent a conventional SDM. This operation requires replacing the cost function with a two-input loop filter, and the selection block with a quantizer, as illustrated in fig. 4.4. If the loop filter (cost function) is now drawn left of the quantizer (selection function), the conventional generic SDM structure of fig. 2.7, repeated in fig. 4.5, results. Note that in the conventional SDM structure the discrete quantization levels of the quantizer block are not explicitly shown, and that the quantizer therefore has only a single input.

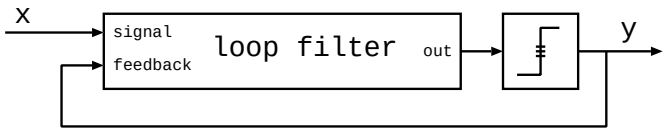


Figure 4.5: Conventional generic model of the Sigma-Delta noise-shaping loop, consisting of 2-input loop filter and quantizer.

In the SDM situation where the cost block is represented by a conventional loop filter, the cost function output is an indication of how well the quantized output signal matches the input signal within a certain bandwidth. The match is calculated over all frequencies, and frequency dependent weighed by the loop filter. The final output of the cost function is a single value that indicates the overall quality of the match, where an in the absolute sense smaller value indicates a better match.

Although the loop filter of an SDM, or any cost function in general,

generates a new output value every clock cycle, this value is not only a function of the current input and current feed-back value, but is also a function of the conversion history. More specifically, previous input and output values are remembered by the filter and influence the filter output. Without this memory function the filter is not able to realize spectral selectivity, and since time and frequency are the dual of each other, a higher frequency selectivity results in a longer time domain signal that influences the filter output. An aggressive filter will have a longer impulse response than a mild filter, and will thus also contain more memory.

4.3 Noise-shaping quantizer with multiple cost functions

The generic noise-shaping structure of fig. 4.3 has a single abstract cost function. In the specific case of an SDM, the cost function translates to a filter. In the general case, since no details of the cost function are specified, the cost function can in principle evaluate a number of quality metrics and generate an output signal on the basis of all the internally calculates scores. The single cost function block can in this case be replaced by a matrix of cost functions. To make this idea more explicit, a generic noise-shaping quantizer will be shown that is able to evaluate two quality metrics, see fig. 4.6.

Since two quality metrics need to be evaluated, two generic cost functions that receive both the input signal and the feed-back signal are required. Furthermore, the two cost functions require direct connections to each other to allow for derived or dependent cost functions, and to send control signals. With two cost functions in place, there will also be two cost function output signals. Since the choice block can only generate a single output value, the two cost values will have to be combined to a single signal that drives the choice block. A third cost function that weighs the two cost values and combines them to a single value is therefore required. This setup where the original single cost function is replaced by a combination of three cost functions is depicted in fig. 4.6 (no control block and control signals shown).

The possibility of evaluating multiple cost functions potentially enables the design of a noise-shaping quantizer with better quality. For example, instead of only evaluating the frequency match as done in a traditional SDM, it is possible to add a second cost function that evaluates the stability of the converter. As long as no stability issues are detected the

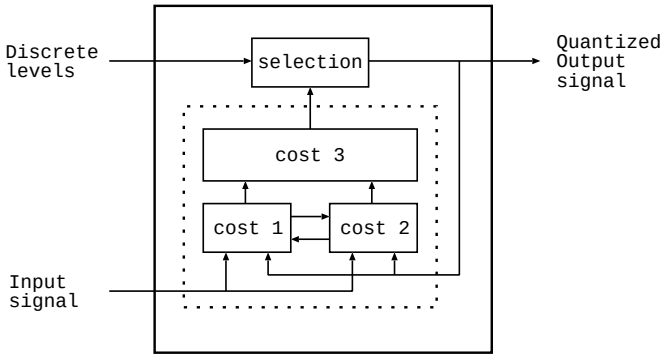


Figure 4.6: Noise-shaping quantizer with three cost functions.

noise-shaping loop will operate as normal, and only the output of the first cost function will influence the output symbol. When the second cost function detects a stability issue, the feed-back loop will change operation and will focus on maintaining stability while relaxing the signal quality aspects.

Another potential use for a second cost function is to improve the transient response quality. In a typical noise-shaping quantizer, the first cost function measures the feed-back quality in the frequency domain. Especially in the case where very high signal-to-noise ratios are desired, the impulse response of the filter is long, i.e. it contains still a significant amount of power after many samples. As a result, the choice of the current output symbol will be influenced by decisions taken in the past and it will influence the future time domain signal. By adding a second cost function, the impact of a symbol on the future can possibly be taken into account and a better transient response can be realized.

4.4 Specific realization structures

Since the noise-shaping quantizer implementation of fig. 4.6 does not make any assumptions on the cost functions, both cost functions *cost 1* and *cost 2* receive three input signals. If the cost functions are less generic, the structure can be simplified.

For example, if both cost functions are independent of each other and require no additional reset or control, a parallel implementation structure is possible, as depicted in fig. 4.7. In this case a third cost function

is required to combine the two output signals into a single signal that can drive the choice block.

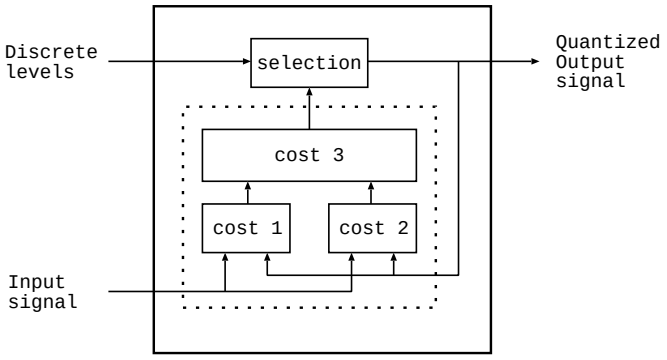


Figure 4.7: Noise-shaping quantizer with two parallel cost functions.

Another special case is possible, where the second cost function only requires the output of the first cost function as an input. In this case a series connection of the two cost function is resulting, and no third cost block is necessary. The resulting block diagram is shown in fig. 4.8.

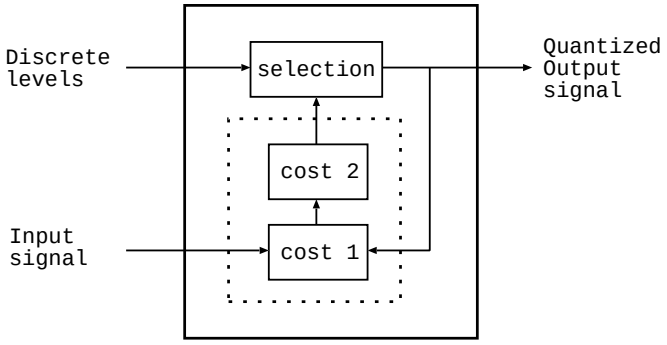


Figure 4.8: Noise-shaping quantizer with two cost functions in series.

Chapter 5

Look-ahead sigma-delta modulation

In this chapter the principles, the benefits, the disadvantages, and the possibilities for realizing a look-ahead SDM are investigated. First, the generic noise-shaping quantizer model from ch. 4 is transformed to a generic look-ahead noise-shaping quantizer (sec. 5.1). This generic model is then mapped to a look-ahead SDM in sec. 5.2. Next, the principle of look-ahead is detailed in sec. 5.3. Since look-ahead requires knowledge about the future, in sec. 5.4 the possibilities for obtaining information about the future are discussed. In sec. 5.5 a basic generic realization of the look-ahead concept, called the full look-ahead algorithm, is shown. On the basis of these concepts a linear model of a look-ahead SDM will be developed in sec. 5.6. Then, the expected benefits and disadvantages resulting from the realization of a look-ahead enabled modulator are presented in sec. 5.7. Although this thesis is focused on DD conversion, the possibilities for performing look-ahead enabled AD conversion are shortly discussed in sec. 5.8. In 5.9 the challenges and possibilities offered by look-ahead sigma-delta modulation in a DD converter are discussed. Finally, conclusions are drawn in sec. 5.10.

5.1 Noise-shaping quantizer with look-ahead

In order to understand how look-ahead can improve SDM performance, the operation of the traditional SDM will be analyzed in more detail. To aid the discussion, the SDM model as shown in sec. 2.2 and reproduced in fig. 5.1 will be used instead of a generic noise-shaping quantizer model.

In a second step the insight will be mapped to the generic noise-shaping quantizer.

Figure 5.1: Generic model of the Sigma-Delta noise-shaping loop, consisting of 2-input loop filter and quantizer.

From ch. 2 it is known that an SDM realizes the shaping of noise with an error minimizing feed-back loop, in which the input signal x is compared with the quantized output signal y (see fig. 5.1). The difference between these two signals is frequency weighed with the loop filter. Differences that have a frequency content which falls outside the signal band are not important, and are contributing little to the output of the filter. If the difference is falling inside the signal band, the quantized signal does not accurately match the input signal. The contribution to the filter output is therefore large. The output of the loop filter is passed to the quantizer, which quantizes the result of the weighing. The quantized signal y is presented as digital output signal and used for generating the feed-back signal. If the SDM is an ADC, the feed-back signal is generated with a DAC which generates an analog representation of the digital output y. In the case of a DD converter the output is directly fed back to the input.

It is clear that if the quantizer would have an infinite high resolution, no error would be introduced in the quantization process. If it is now also assumed that the loop-filter output is only resulting from in-band errors, the feed-back signal will cancel those errors exactly if there is no delay in the loop. In reality, the quantizer has a finite resolution and the loop-filter output is not only resulting from in-band errors. Furthermore, there is always a delay in the loop of at least one clock cycle. As a result, the feed-back signal will not cancel the in-band errors exactly and quantization noise is added. The continuous feed-back of the weighed error signal, results in the shaping of the quantization noise. Key in this process is the function of the loop filter, which forms a memory-based element and therefore enables de-correlation of the quantization noise.

From the above discussion it can be concluded that if the power in the filter output becomes less, the quantization noise has less in-band frequency content. In other words, the input signal is better approximated

and the quantization noise is pushed further out of band. In the conventional SDM algorithm it is not the power of the filter output that is measured and minimized directly, but the feed-back strategy attempts to regulate the filter output towards zero. In the special case of a 1-bit modulator this operation achieves the same effect and as a result a 1-bit SDM attempts to minimize the power of the filter output.

If the Sigma-Delta modulation algorithm would be able to always select the optimal output symbol, i.e. the symbol that results in a minimal filter output energy instead of minimal instantaneous output power, the operation of an SDM could be improved. More specifically, the feed-back approach cannot guarantee that the selected output symbol will minimize the error energy because the loop filter only evaluates the current situation, and based on this outcome it is attempting to change the future in a beneficial way. By changing the noise-shaping loop to actively search for and determine what output symbol delivers the best result, it should be possible to improve the performance.

With a traditional SDM it is not possible to determine what output symbol will result in the lowest error energy, since the cost function (loop filter) has only knowledge of (part of) the input and the output signal. The cost function has no knowledge of what possible output symbols can be selected. This situation is also clearly recognizable in the generic noise-shaping quantizer model (sec. 4.2). Only when the cost function is aware of what output symbols (discrete levels) are available, it is possible to evaluate the effect of the different output values and take their impact into account. With this additional information the structure would thus be able to effectively look ahead in time compared to the feed-back approach.

A generic noise-shaping quantizer with the possibility of performing look-ahead modulation is depicted in fig. 5.2. Comparison with the original noise-shaping quantizer (fig. 4.3) reveals that the only difference is the additional connection of the discrete levels to the cost function. However, because of this additional input the internals of the cost function can be very different than those of the traditional noise-shaping quantizer.

5.2 Look-ahead enabled SDM model

In sec. 4.2 it was demonstrated that the SDM model could be obtained from the generic noise-shaping quantizer model, by simply changing the naming of the blocks. The same procedure can be applied to the gen-

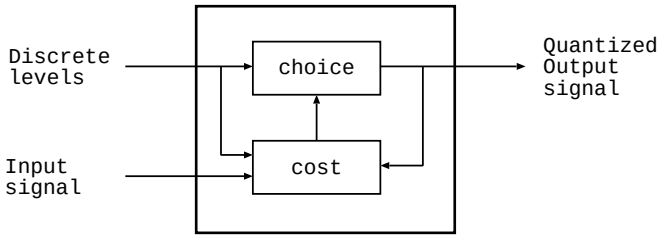


Figure 5.2: Look-ahead enabled noise-shaping quantizer.

eric noise-shaping quantizer with look-ahead capabilities of fig. 5.2 to generate a look-ahead enabled SDM.

The cost block is made explicit by naming it look-ahead loop filter, and the choice block is named look-ahead quantizer. Both blocks receive the set of discrete levels that are available to the modulator. The resulting structure, with the loop filter placed at the traditional location left of the quantizer, is depicted in fig. 5.3.

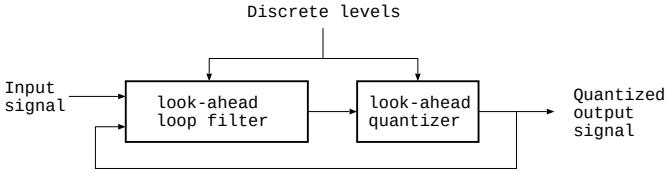


Figure 5.3: Look-ahead enabled SDM.

Compared to a normal SDM there are two main differences visible. First, the loop filter has knowledge of what quantization levels are supported. With this knowledge it can evaluate what the impact is of one or possibly a series of output symbols. Since there are in every SDM at least two symbols to choose from, the look-ahead loop filter will generate a multitude of output signals, i.e. one output for every possibility. Second, the quantizer in a normal SDM translates the loop-filter output in an output symbol. In the case of a look-ahead SDM, the quantizer will have to translate the combination of loop-filter outputs to a single output symbol. The look-ahead quantizer thus requires more than one input signal, and by applying a cost function to the input signals it determines which solution is the best, and translates this to an output symbol.

From the above it can be concluded that the look-ahead quantizer of

fig. 5.3 effectively consists of two concatenated functions, i.e. a cost function that receives many signals and generates a single output, and an output function that maps this signal to an output symbol. The cost function determines which of the loop-filter outputs is the best signal, e.g. by measuring which signal has the lowest energy or the smallest maximum amplitude. The subsequent output function will translate this result to the final converter output symbol, and will also communicate the selection result back to the look-ahead loop filter.

5.3 Look-ahead principle

The main idea of the look-ahead algorithm is to obtain information about the future, and to use this information in a beneficial way. More specifically, in the case of an SDM the future input signal will be used to improve the conversion result. All other SDM signals, i.e. the loop-filter states and the output signal, are derived from the input. Therefore, if the future input signal is known, the future loop-filter states and future output codes can be generated. By examining these generated future signals, possible problems that will arise if the conversion process continues without any change can be detected. This enables the possibility to influence the current SDM operation and circumvent the problem.

Obviously, if more future data is available, it is possible to look ahead further. However, not necessarily a lot of future data is required for detecting conversion problems. For example, by looking at how the loop-filter states develop over a short period of future time, a nearing instability can be detected. However, since the point of overload is already reasonably close, drastic measures, e.g. temporarily reducing the loop-filter order, are required to keep the modulator stable.

If the amount of look-ahead would be larger, instability could be detected further in advance and more subtle measures could be taken. An example reaction could be to change the next quantizer output, i.e. overrule the decision and change it to a symbol which is less likely to cause instability. However, there is no guarantee that this action will actually solve the problem, since the actual effect on the future is not evaluated a-priori. Only in the future conversion cycle or cycles it will be known if the future problem has been solved. Therefore, an approach that actively tests for the (future) quality of a solution instead of this passive approach is preferred.

In the active approach, a search for the best sequence of output codes is performed, such that this knowledge can be used in the main conversion

process. As a result of this selection process, the quality of the main conversion can be higher. In order to perform the search for the best sequence of output symbols, the operation of the SDM is changed from the normal feed-back mode, in which the quantizer output is based on the loop-filter output, to a mode in which the quantizer output is forced to a pre-determined value.

This control over the SDM quantizer output is effectively realized by the schematic of fig. 5.4, which shows the details of a possible implementation of a look-ahead enabled SDM. Clearly recognizable in this specific implementation is the look-ahead loop filter, the look-ahead quantizer, and the feed-back from the output to the filter. The look-ahead loop filter, indicated by the dashed box, consists of a look-ahead filter (LA filter), a sequence generator, and the traditional SDM loop filter. The output of the look-ahead filter is passed to the look-ahead quantizer which generates the quantized output signal and provides the feed-back signal. Thus, the structure basically equals a traditional SDM in which the loop filter determines how the quantization noise is shaped, complemented with additional look-ahead logic.

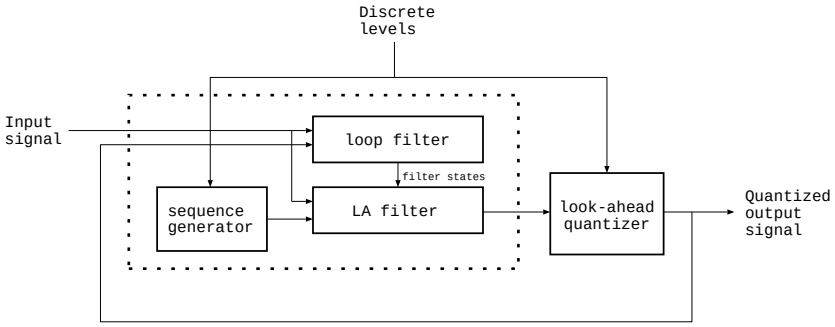


Figure 5.4: Look-ahead enabled SDM.

The functioning of the structure is as follows. The main loop filter receives the current input signal and the feed-back signal, and updates the internal filter states on the basis of these inputs. The filter states of the main loop filter are passed to the look-ahead filter, where they are loaded. The sequence generator block will now provide symbol sequences to the look-ahead filter, which processes these in combination with the future input signal. The look-ahead filter output is passed to the look-ahead quantizer, which evaluates a quantizer cost function to select the best symbol sequence. As a last step, the quantized output signal is

determined from this sequence.

For example, for a look-ahead depth of two time steps, there are four possible feed-back signals to evaluate for a 1-bit quantizer. These four different feed-back signals will result in four different filter responses. Subsequently, from the set of possible responses the best signal is selected. The selection is realized by evaluating the cost function.

Finally, once the sequence of symbols with the lowest cost has been determined, the main conversion process can continue. The outcome of the selection operation is a sequence of symbols with a length equal to the look-ahead depth. From this sequence typically only the first symbol is used as the output signal, although in theory more symbols could be used to speed up calculations at the cost of a reduced look-ahead depth. The rest of the sequence is discarded, as it was only required to discover what combination of symbols matches best with the input signal over the length of the look-ahead depth. The output value, or the selected sequence in a more general case, is passed back to the loop filter. In the next conversion cycle the same process will be repeated, again evaluating all the sequences in order to decide on the optimal feed-back symbol.

The approach of selecting the feed-back symbol based on the knowledge from look-ahead can be realized in multiple ways. Because of the much improved performance over passive look-ahead approaches, in this thesis only look-ahead approaches that actively test and evaluate the quality of future responses are considered.

5.3.1 Quantizer cost function

Each of the look-ahead filters receives N samples from the future input signal in combination with a sequence of N trial feed-back symbols, and generates in response a sequence of N output values. The quality of these output signals is measured in the look-ahead quantizer. This measurement is realized by evaluating a quantizer cost function, i.e. a second cost function that operates on the output signal of the loop filter, which forms the first cost function.

The objective of the quantizer cost function is to first translate the sequence of look-ahead loop-filter output values to a single value that represents the quality of the applied feed-back sequence. In a second step the overall best result is selected. The selected feed-back sequence is then used to generate the final output symbol or symbols.

Depending on the selected quantizer cost function, different types of optimization can be performed. For example, it is possible to optimize

for minimal filter output energy, measured over a fixed time period. This would result in the cost function

$$C = \sum_{i=0}^{N-1} w^2(i) \tag{5.1}$$

where $w(k)$ represents the look-ahead filter output signal.

With

$$c(k) = w^2(k) \tag{5.2}$$

this can be rewritten as

$$C = \sum_{i=0}^{N-1} c(i) \tag{5.3}$$

which can be simplified to

$$C(k) = C(k-1) + c(k) \tag{5.4}$$

Since this cost function minimizes the in-band error signal, it offers the potential to reduce signal distortion.

Another possible optimization criterion could be to minimize the maximum value of the filter output. This can be accomplished by assigning the maximum value as the final cost:

$$C = \max(|w(i)|) \tag{5.5}$$

This cost function could possibly improve the stability of the converter. Since the maximum filter output is reduced, most likely also the internal filter signal swing is reduced, and this reduces the chances of overloading the modulator.

In principle there is no restriction on the cost function, and any linear or non-linear combination of functions and variables can be used. In this thesis, however, in virtually all cases the optimization criterion will be to achieve minimal energy at the output of the weighing filter, as realized by eq. 5.1.

5.4 Obtaining information about the future

Key in the idea of look-ahead modulation is the use of the future input signal. In the discussion of sec. 5.3 it was implicitly assumed that this future signal is available. In general, however, the future input signal

is not available and special action has to be taken in order to obtain it. Ideally, the information about the future should be 100% accurate, but also an estimate or approximation of the future might be used to an advantage.

5.4.1 Approximated future input

For an interpolative SDM an estimation of the future input signal can be obtained without any difficulty. Because of the high oversampling rate, the input signal can be approximated as a constant for the next clock cycles. Therefore, an approximation of the future SDM response can be derived by assuming that the input signal will not change.

Because of the predictive nature of this algorithm we denote the technique as predictive look-ahead. The disadvantage of predictive look-ahead is that, as the name implies, it is only a prediction of what will most likely happen and it could therefore be incorrect, possibly resulting in a sub-optimal decision. Obviously, the further ahead the future is predicted, the larger the prediction error will be. Still, for predicting the near future such an approximation might prove useful.

For example, in [2, 60] a digital Look-Ahead Decision feed-back SDM is published that decides on its feed-back value under the assumption of a constant input signal. The quantizer output is chosen such that it minimizes the magnitude of the future internal integrator values. It is reported that minimization of the weighed integrator states results in an, on average, smaller quantizer input, which results in increased stability and higher SNR.

5.4.2 Actual future input

Instead of using a prediction of the input signal, it is possible for a real-time system to use the actual future input signal, without violating the rules of causality. In order to realize this, it is required to change the reference point that determines what is 'now' and what is the 'future', i.e. by denoting a time moment in the past as 'now', the actual 'now' becomes the 'future'.

This shift of the reference point can be realized by delaying a signal, as illustrated in fig. 5.5. The continuous-time signal $x(t)$ is sampled with sampling period T, resulting in the discrete-time signal $x(nT)$. The signal $x(nT)$ is passed to a delay line which is clocked at the same rate as the sampling rate. As indicated below the delay line, from left to

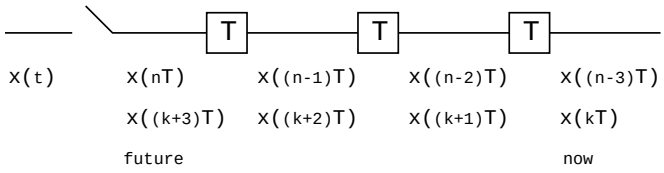


Figure 5.5: A change of the reference point enables the use of 'future' data.

right the samples $x(nT)$ are from earlier time moments. When signal $x(t)$ is sampled (sample $x(nT)$), the oldest value in the delay line is from three sampling periods earlier (sample $x((n-3)T)$). If index n is replaced by $k+3$, the oldest sample becomes $x(kT)$ and the newest sample $x((k+3)T)$. Effectively, the index change from n to $k+3$ has changed the reference point such that the oldest sample becomes 'now' and the most recent sample is placed in the 'future'.

By tapping the delay line at intermediate locations, every sampled value between 'now' and the 'future' is available. The length of the delay line determines how much future data is available. At the same time every additional sample delay increases the latency of the system by a clock cycle, since all operations are performed on the delayed version of the signal. With this technique it is effectively possible to look into the future at the cost of a latency that is equal to the amount of look-ahead.

Not in all situations real-time SDM operation is required, e.g. creation of the final masters for SA-CD is often an off-line process and is allowed to take more time. In those situations the complete signal is typically first recorded and stored in a multi-bit format, and the conversion to a 1-bit format is done afterwards. During this conversion in principle the complete signal is available, and future data can now be accessed and used directly by the look-ahead process.

5.5 Full look-ahead algorithm

The principle of look-ahead in which the full solution space is investigated, is known under several names and several different approaches to implement the algorithm have been demonstrated in literature. For example, in [18] the principle is called receding horizon quantizer. In [6] two specific implementations of full look-ahead are shown, called full-tree algorithm and stack algorithm. In all these approaches, although

implemented differently, the objective is to decide upon the next output symbol on the basis of the quality of the set of all possible future symbols.

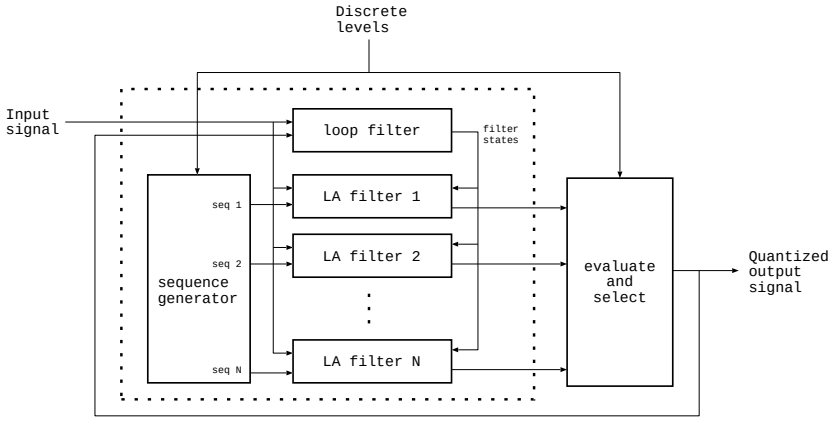


Figure 5.6: Parallel realization of a look-ahead SDM.

In a parallel realization of the full look-ahead algorithm, as depicted in fig. 5.6, the process of evaluating the quality of a series of future symbols is as follows. After the main loop filter has processed the current input and the current feed-back signal, the internal filter states are transferred to the N look-ahead filters (LA filters). Each of these filters will now receive a different trial sequence, such that all possible output sequences are compared with the input signal. In the case of a 1-bit converter this results therefore in $N = 2^M$ sequences, each with a length of M symbols.

By applying the future M values of input signal $x(k)$ and a set of trial feed-back symbols $fb(k)$ to a look-ahead filter, M filter outputs $w(k)$ are obtained. On each of these M output values a cost function, e.g. $c(k) = w^2(k)$, is applied in order to obtain the cost for applying symbol $fb(k)$. This processing path, starting at the look-ahead filter input, is depicted in fig. 5.7 for a feed-forward loop filter.

From the signal $c(k)$ the accumulated cost $C = \sum_k c(k)$ is next calculated and recorded. This process is performed in parallel for each of the 2^M trial sequences. Finally, the solution with the lowest accumulated cost is selected, and from this the next output symbol is determined, i.e. the first symbol of the best solution is selected as the output symbol. This symbol is now applied as actual feed-back signal in order to advance time.

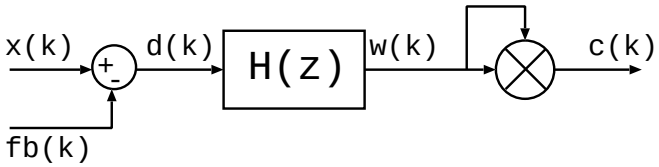


Figure 5.7: Look-ahead feed-forward filter structure with concatenated cost function for evaluating the quality of trial feed-back values.

As an example, the process for a look-ahead depth of two will be demonstrated.

Since the look-ahead depth is two, there are four sequences possible. All of these sequences will result in a certain cost. The best sequence, i.e. the sequences with the lowest cost, will now determine what output symbol will be selected. If the best sequence started with a '0' the next output symbol will be a '0', otherwise it will be selected as '1'. Two of the four sequences will start with the selected symbol, but it is not necessary that these two sequences are the best and second best solution. In fig. 5.8 the cost associated with expanding an example bitstream with two bits is depicted.

The cost for adding the zero or one bit to the sequence is printed next to the branch. The accumulated cost of the total sequence is printed at the vertices. In this example, the sequence "10" results in the lowest accumulated cost of 0.2, therefore the decision for the next symbol will be a '1'. Note that the second best sequence is "01", which would result in a '0' symbol instead.

In the next time step again four possibilities are evaluated, following the '1' bit of the previous clock cycle. This situation is depicted in fig. 5.9. In the new situation the path with the lowest accumulated cost is "00", therefore the bitstream is continued with a '0'.

By comparing fig. 5.9 with fig. 5.8 it can be recognized that the cost for the first one respectively zero bit, is identical to the cost that was calculated in the previous clock cycle for the last one and zero bit of the selected solution. This result is exactly as expected, since the new solutions space overlaps with the previous examined solution space. It can be seen that with a look-ahead of N symbols there is an overlap of $N-1$ symbols. In the case of a 1-bit converter, out of the 2^N results to calculate, 2^{N-1} results have been calculated already in the previous clock cycle. Therefore, by reusing previously calculated results a reduction in

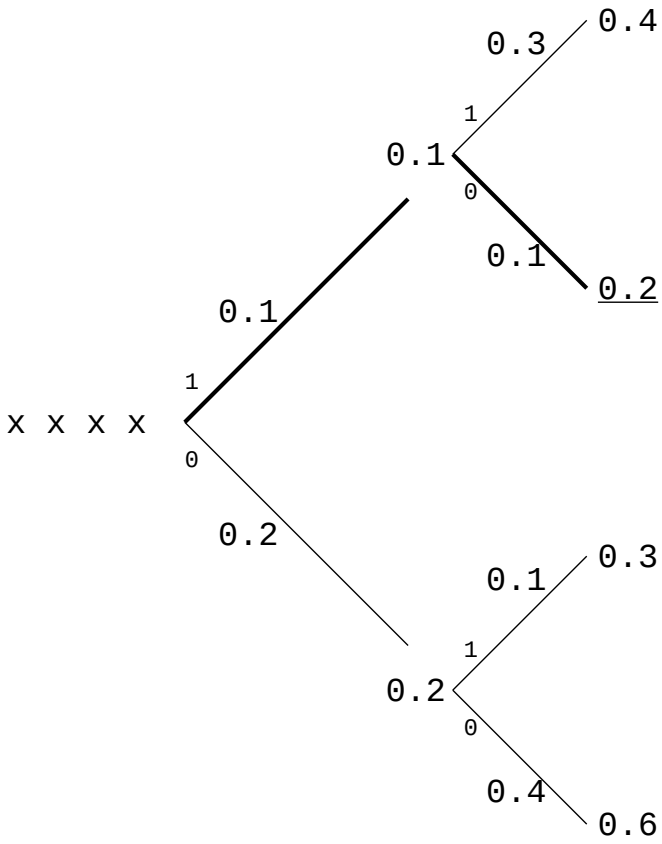


Figure 5.8: Four possible sequences for expanding the example bitstream (clock cycle 1). The cost for adding a bit is printed above the branch, the accumulated cost at the vertices. The solution with the lowest cost is indicated with thick branches and has the cost value underlined.

the computational complexity can be realized.

5.6 Linear modeling of a look-ahead SDM

From a high level perspective a look-ahead SDM is very comparable to a normal SDM. However, the realized signal transfer and noise transfer are different because of the different feedback strategy. In order to see the

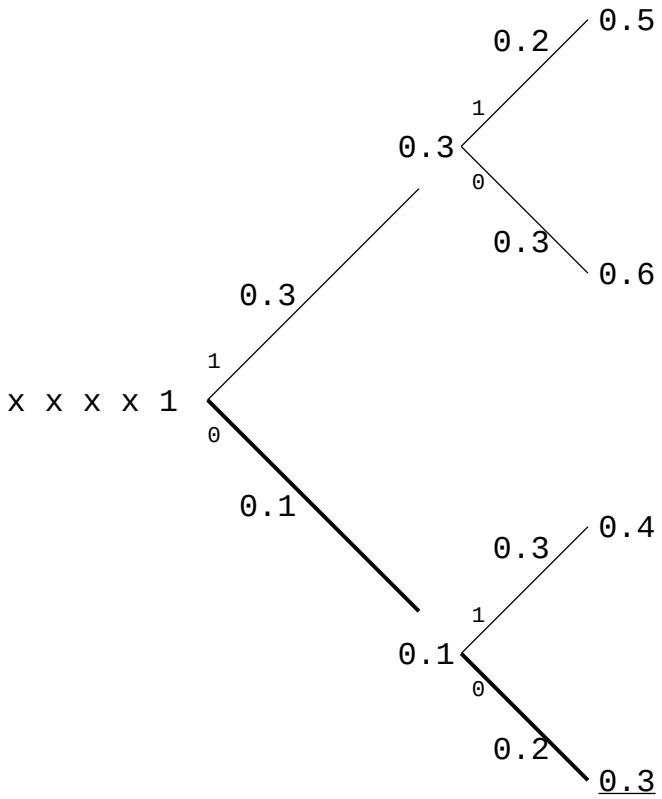


Figure 5.9: Four possible sequences to continue the bitstream in the second clock cycle. The cost for adding a bit is printed above the branch, the accumulated cost at the vertices. The solution with the lowest cost is indicated with thick branches and has the cost value underlined.

impact of this different strategy, a linear transfer model for predicting the STF and NTF of a look-ahead SDM will be derived and compared with those of a normal SDM.

5.6.1 Boundary conditions and assumptions

For the derivation of the linear model it is assumed that the look-ahead SDM is ideal, such that the optimal feed-back sequence will be found. Under this assumption it holds that, for the feed-forward look-ahead

structure of fig. 5.7, signal $fb(k)$ will be selected such that minimal error energy will result, i.e. $\sum_k c(k)$ will be minimal. Since $H(z)$ is low-pass and input signal $x(k)$ is assumed to have all its energy in the pass-band of $H(z)$, feedback signal $fb(k)$ will equal signal $x(k)$ for low frequencies and have minimal residue energy in the pass-band. Thus, signal $d(k)$, which equals the difference between signal $fb(k)$ and input signal $x(k)$, will not contain any signal components and will have a minimal amount of noise components in the pass-band of $H(z)$.

The cost $c(k)$ for residue energy at higher frequencies is increasingly less as the attenuation of filter $H(z)$ increases. The optimization process strives for minimal $\sum_k c(k)$, which is achieved when $d(k)$ has most of its energy at high frequencies. Ideally, $d(k)$ would thus be a tone at $Fs/2$, but such a signal can only exist for the situation $x(k) = 0$ (or when $x(k)$ is equal to a repetitive $+1, -1$ sequence). For every non-zero input signal the spectrum of $d(k)$ will be noise alike. Under this assumption that $d(k)$ resembles noise (i.e. is not a correlated signal), minimal filter output energy is achieved when $d(k)$ has a noise density which is rising with frequency, i.e. a high noise density for frequencies close to $Fs/2$.

From prediction filtering it is known that when the prediction filter has the inverse spectral shape of the input signal, the filter output has minimal energy and has a flat frequency distribution. In the look-ahead sigma-delta modulation algorithm it is not the filter that is designed to match the signal, but the signal is designed to match the filter. The result of this 'inverse' optimization is identical to the result obtained for prediction filtering, and the resulting output signal $w(k)$ is white.

With $w(k)$ white it holds that

$$D(z) \cdot H(z) = T \tag{5.6}$$

with $D(z)$ the frequency distribution of the difference signal $d(k)$, $H(z)$ the filter transfer, and T a constant.

Therefore, the shape of $D(z)$ is the inverse of $H(z)$ multiplied by a constant gain T:

$$D(z) = \frac{T}{H(z)} \tag{5.7}$$

However, in practice the energy of signal $d(k)$ is not constant since it is a function of the energy of the input signal, and as a result eq. 5.7 will not hold but can only be approximated. With this in mind, eq. 5.7 will be used for deriving the *linearized* NTF of a look-ahead modulator.

5.6.2 Feed-forward look-ahead SDM

For the feed-forward look-ahead SDM of fig. 5.7 the linearized NTF and STF equations are derived.

NTF

The NTF of a feed-forward look-ahead SDM is obtained by solving eq. 5.7 for the situation of no input signal, i.e. $x(k) = 0$. With $x(k)$ equal to zero, signal $d(k)$ will be equal to $-fb(k)$. In practice the situation of no input signal would result in a high frequency limit cycle, but if it is assumed that the output spectrum is consisting of shaped noise, the NTF is found to be:

$$NTF_{\text{LA SDM}}(z) = \frac{T}{H(z)} \tag{5.8}$$

$$\propto \frac{1}{H(z)} \tag{5.9}$$

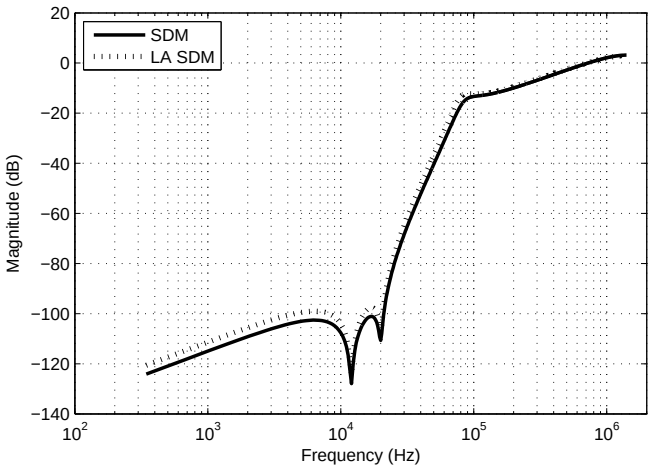


Figure 5.10: Linearized NTF for a normal SDM (effective quantizer gain of 2.0) and a look-ahead SDM, scaled to have the same power gain for a flat input frequency distribution.

Thus, the linearized NTF predicts a shaping of the quantization noise which is proportional to the inverse of the loop filter H, whereas for

a normal SDM the shaping is proportional to $\frac{1}{1+H}$. At first sight the impact on noise suppression might seem negligible since $|H(z)|$ is large for low frequencies. However, for high frequencies a significantly different noise transfer is realized. In order to compare the two transfers they need to be normalized to have the same power gain. Since the gain is a function of the input frequency, the total gain can only be made equal for an assumed input frequency distribution, e.g. flat over the complete frequency span. Furthermore, in the case of the normal SDM the assumed effective quantizer gain influences the shape and power gain of the NTF. In the case of the look-ahead SDM there is no effective quantizer gain to take into account and the shape of the NTF is fixed. As a result, the two NTF curves can move with respect to each other, depending on the assumed effective quantizer gain of the SDM.

In fig. 5.10 the predicted NTFs are compared for an assumed effective quantizer gain of 2.0 and a flat input frequency distribution. Under theses assumptions a higher baseband noise-floor is predicted for the look-ahead SDM, while at very high frequencies a lower noise level is predicted for the look-ahead SDM. Since the power gain of both NTFs is equal, it is expected that the SNR of the look-ahead SDM will be slightly lower than that of the normal SDM.

STF

The STF of a feed-forward look-ahead SDM can be found by solving for a zero cost output of the look-ahead filter structure, which is equal to solving for a zero filter output. This results in solving the equation:

$$(x - fb) \cdot H(z) = 0 \tag{5.10}$$

Which can simplified to:

$$x - fb = 0 \tag{5.11}$$

$$x = fb \tag{5.12}$$

The STF of a feed-forward look-ahead SDM therefore equals:

$$STF_{\text{FF LA SDM}}(z) = 1 \tag{5.13}$$

According to this prediction the signal is perfectly encoded without any frequency dependent gain or phase shift. Compared to a normal SDM, the signal transfer behavior is almost identical for low frequencies. For frequencies far out of band a different behavior will be observed since the look-ahead SDM will not show a roll-off.

5.6.3 Feed-back look-ahead SDM

For the example second order feed-back loop filter of fig. 5.11 the specific linearized NTF and STF equations for a look-ahead SDM will be derived, as well as generic expressions for nth order feed-back look-ahead modulators.

NTF

In the case of a normal SDM the NTF of a feed-back and a feed-forward filter can be made equal by using the same coefficient values. This relation also holds for a look-ahead SDM, since the transfer from the fb input to the output c is identical for both structures. Thus, the NTF of eq. 5.9 is also valid for a feed-back look-ahead SDM:

$$NTF_{\text{LA SDM}}(z) \propto \frac{1}{H(z)} \tag{5.14}$$

STF

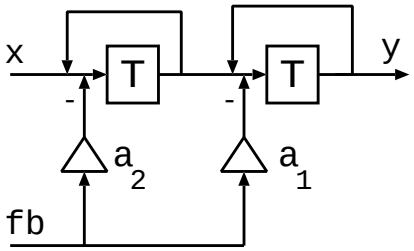


Figure 5.11: Second order feed-back Sigma-Delta filter

For the second order feed-back filter of fig. 5.11 the STF is found by solving:

$$\big((x - fb \cdot a_2)\frac{z^{-1}}{1 - z^{-1}} - fb \cdot a_1\big)\frac{z^{-1}}{1 - z^{-1}} = 0 \tag{5.15}$$

Which can be rewritten as:

$$(x - fb \cdot a_2)\frac{z^{-1}}{1 - z^{-1}} - fb \cdot a_1 = 0 \tag{5.16}$$

$$fb \cdot (a_2\frac{z^{-1}}{1 - z^{-1}} + a_1) = x\frac{z^{-1}}{1 - z^{-1}} \tag{5.17}$$

The STF is now given by:

$$\begin{aligned} STF &= \frac{fb}{x} \\ &= \frac{\frac{z^{-1}}{1-z^{-1}}}{a_2\frac{z^{-1}}{1-z^{-1}} + a_1} \\ &= \frac{z^{-1}}{a_2 \cdot z^{-1} + a_1 \cdot (1 - z^{-1})} \\ &= \frac{z^{-1}}{a_1 + (a_2 - a_1) \cdot z^{-1}} \end{aligned} \tag{5.18}$$

For a normal second order feed-back SDM with the loop filter of fig. 5.11 the STF equals:

$$STF = \frac{z^{-2}}{1 + (a_1 - 2) \cdot z^{-1} + (a_2 - a_1 + 1) \cdot z^{-2}} \tag{5.19}$$

which can be written in a generic form as:

$$STF_{\text{FB SDM}} = \frac{z^{-p}}{D + N} \tag{5.20}$$

where D equals $(1 - z^{-1})^p$.

For a generic (higher order) look-ahead SDM with feed-back loop filter H' the STF can be found by solving:

$$x \cdot I^p - fb \cdot H' = 0 \tag{5.21}$$

where p denotes the filter order, and I is an integrator:

$$I = \frac{z^{-1}}{1 - z^{-1}} \tag{5.22}$$

Filter $H' = \frac{y}{fb}$ is of the form:

$$H' = \frac{N}{D} = \frac{N}{(1 - z^{-1})^p} \tag{5.23}$$

where N and D are the numerator respectively denominator of the ori-

ginal SDM filter H. Substitution and simplification leads to

$$\begin{aligned} STF_{\text{FB LA SDM}} &= \frac{fb}{x} \\ &= \frac{I^p}{H'} \\ &= \frac{z^{-p}}{(1-z^{-1})^p} \cdot \frac{(1-z^{-1})^p}{N} \\ &= \frac{z^{-p}}{N} \end{aligned}$$

Comparison of eq. 5.20 with eq. 5.24 reveals that the linearized STF of a feed-back look-ahead SDM differs slightly from that of a normal SDM.

In fig. 5.12 the linearized STFs of a look-ahead modulator and normal SDM in feed-forward and feed-back operation are compared. The loop filter is a 5th order Butterworth filter with a corner frequency of 100 kHz and two resonator sections at 12 kHz and 20 kHz. The transfer realized by the feed-back modulators is very comparable, both showing a limited amount of gain around the corner frequency of the filter and then a strong fall-off. In the case of a feed-forward modulator a perfect unity transfer is realized for the look-ahead SDM. The feed-forward SDM shows the strongest peaking of all the modulators, i.e. approximately +8 dB of gain just above the filter corner frequency, before falling off first order.

5.7 Benefits and disadvantages of look-ahead

The potential advantages and disadvantages of using look-ahead in a noise shaping converter will be discussed separately.

5.7.1 Benefits

The potential benefits of a look-ahead modulator over a normal SDM consist at least of an improvement of the converter's stability, an increase in the linearity, and the realization of an improved transient response. A detailed description and motivation of these improvements follows.

Improved stability

One of the expected benefits of a look-ahead enabled SDM, is an increase in the converter's stability. This increase can be realized because

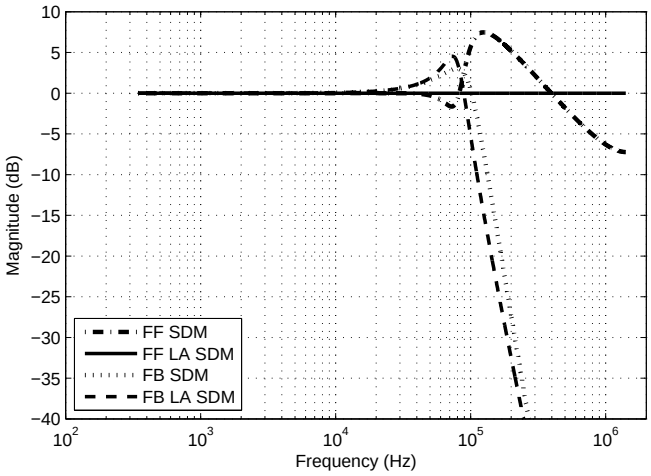


Figure 5.12: Linearized STF of a feed-forward and a feed-back modulator for a normal SDM and a look-ahead SDM. The loop filter is a 5th order 100 kHz Butterworth filter with resonators at 12 kHz and 20 kHz.

the look-ahead algorithm will reject bit sequences that have a bad match with the input signal. More specifically, patterns that will result in instability will typically first cause a filter response which deviates significantly from the input signal. Thus, long before the system reaches instability large error signals will result. The look-ahead algorithm will force the selection of patterns that cause, on average, the smallest error. As a result, the system will not as easily become unstable and the input level for which instability occurs will be increased. A limited look-ahead depth should be enough to realize a reasonable increase in input range. With an infinite look-ahead depth it should be possible, in theory at least, to realize a 100% modulation depth independent of the number of quantization levels and loop-filter order. In the case of 1-bit encoding, however, the number of bit sequences that can accurately describe the input signal at such high modulation levels strongly decreases. Therefore, a decrease in SNR performance for those high levels is most likely to occur. In practice, however, no results of such a strong increase of input stability have been published. The largest increase in stability is reported by the author in [32], where an increase from 66% maximum modulation depth to 88% modulation depth is demonstrated for a fifth order modulator. In concurrent work also stability increases are reported, but typically no results on the amount of increase are reported. For

example, in [38] an increase in the useable input range is demonstrated, the exact amount of increase achieved is not quantified however. In [25] it is shown that a more aggressive noise-shaping filter can be stabilized by means of look-ahead, but again no quantification is made of the amount of increase. Also in [21, 22] an increase in input range of a 1-bit modulator is reported, realized by step-back instead of look-ahead.

Increase in linearity

Another potential benefit of look-ahead modulation, is the realization of an increase in linearity. Distortion, introduced by non-linearities in the SDM loop, is reduced by the loop-gain of the modulator. However, because of finite loop-gain not all errors can be suppressed, and even perfectly implemented digital modulators, especially 1-bit, suffer from harmonic distortion. There are several options to attenuate these unwanted components, namely increasing the loop-filter order, making the loop filter more aggressive, and adding dither. In the case where the loop filter is made more aggressive, i.e. a higher corner frequency, the gain of the filter is effectively increased, causing a reduction of the distortion components. If instead the loop-filter order is increased, the gain of the filter also becomes higher for the same corner frequency. If the corner frequency is now changed to cancel the increase in gain, still fewer distortion components are present than in the original situation, since the higher order filter introduces more randomization in the bit-stream, reducing tonal behavior. However, the disadvantage of the higher order filter is an increase in circuit complexity and cost, power consumption, as well as a reduction of the stability of the converter. As an alternative it is also possible to dither the converter. This will also cause break-up of repetitive patterns, at the cost of a decrease in SNR. Depending on how much dither is required for removing the tones, also the stable input range might be reduced.

Since the distortion components are not present in the input signal but only in the feed-back, a look-ahead modulator should in theory be able to detect and avoid those erroneous components. In the evaluation of the possible feed-back patterns, the cost of patterns that cause distortion components will be higher than that of patterns that do not introduce distortion. However, in order to detect such components, a large look-ahead depth might be required. More specifically, the distortion tones that need to be detected, are present in the pass-band of the converter. The time domain sequence describing the shortest period of these tones has therefore a length, expressed in number of samples, of at least two times the oversampling ratio. In practice, this means that in a 64 times

oversampled audio system the shortest sequence (20 kHz tone) consists of 141 bits. A 1 kHz tone has a length of approximately 2800 samples. In order to detect such a low frequency tone not the complete period is required, but at least a fraction of it. It is unclear how big this fraction should be. If the look-ahead depth is large enough, not only high frequency but also low frequency distortion components will be detected and suppressed.

Improved transient response

In ch. 3 it was concluded that the non-steady-state signal conversion performance of a normal SDM, measured in terms of the SINAD, is equal to the steady-state conversion performance, i.e. there are no significant encoding errors introduced because of the dynamically changing input signal. However, experiments with various modulator structures have indicated that not all structures sound the same. In these experiments, conducted under controlled circumstances in professional recording studios, several differently encoded versions of the same music material where presented to audio professionals, and they where asked to describe the qualities of the sound. The outcome of these experiments was that the material that was encoded with the feed-forward look-ahead enabled modulators had a 'better defined' sound than the normal SDM encoded versions of the same material. In this context 'better defined' reflects to a better phase behavior, which suggests an improved transient response. Thus, although SINAD measurements on a normal SDM show no degradation for non-steady-state signals compared to steady-state signals, listening experiments indicate that a look-ahead enabled SDM realizes an audibly better encoding quality.

A possible explanation is the following. In a noise shaping converter the encoding error is evaluated in the baseband only, i.e. in the pass-band of the loop filter. The more aggressive this loop filter is, the better the separation between the baseband and the quantization noise is, and the higher the resulting SNR will be. However, a more aggressive filter, i.e. more frequency resolution, results in a longer impulse response and therefore less time resolution. Since time resolution and frequency resolution scale inversely of each other, it is not possible to have both a good time and frequency resolution with a normal SDM. A look-ahead modulator on the other hand, can evaluate the impact of a feed-back symbol on the future. Thus, by looking into the future, the reduction in time resolution caused by the filter can be partially restored, thereby improving the transient response. Because these improvements are very subtle, there is no clearly measurable effect on the SINAD value, but

the effects can still be audible.

Another possible cause for a less than optimal transient response is the linear distortion that is added by an SDM. Although linear distortion does not influence the SINAD, it does alter the time domain shape of the signals that are encoded. Since audio signals are non-steady-state, and are consisting of a broad range of frequencies which are simultaneously present, the frequency dependent phase shift effectively realizes a small time shift between the different frequency components. Such a frequency dependent group delay is not desirable, and is known to cause an audible reduction of the signal quality. From the linear modeling of a look-ahead SDM (sec. 5.6) it is known that the STF of a feed-forward look-ahead SDM is predicted to be equal to a unity transfer. As a result, a feed-forward look-ahead SDM should have a constant group delay, which could explain the higher perceived audio quality.

5.7.2 Disadvantages

From sec. 5.6 it is known that, compared to a normal SDM with identical loop filter, a look-ahead SDM will realize a slightly lower SNR. Because of the increased stability of a look-ahead SDM it should be possible to compensate for this decrease by using a more aggressive filter, effectively canceling this disadvantage. However, if it is not possible to compensate for the reduction in SNR, this reduction is a serious disadvantage of the look-ahead approach.

Further disadvantages of the look-ahead approach are an increased circuit complexity, a higher power consumption, and a higher realization cost, i.e. more silicon area or more CPU cycles per second. More specifically, since a large number of solutions has to be evaluated at every clock cycle, the power consumption of the converter will increase. The increase is expected to scale slightly faster than linear with the number of solutions to evaluate. Furthermore, in order to perform all calculations in real time, parallel hardware realizations or high clock frequencies will be required. The realization cost of the circuit will thus also increase significantly. Since the number of solutions to investigate doubles with every additional sample of look-ahead for a 1-bit SDM, and even grows faster for multi-bit quantizers, the aforementioned increases can easily become a factor hundred to thousands, enforcing a practical limit on the realizability of the concept. A last disadvantage of the look-ahead approach is the increase in system complexity. Large numbers of signals and parallel circuits will have to be managed, increasing the probability of introducing errors in the implementation.

5.8 Look-ahead AD conversion

In this thesis the focus is on efficient look-ahead digital-to-digital sigma-delta modulation with a special focus on application to Super Audio CD. Still, it is of interest to know if there would be advantages in realizing a look-ahead enabled analog-to-digital SDM. Potential issues in the realization of the analog circuits are briefly examined. An alternative solution that combines a traditional ADC with a look-ahead enabled DD converter is also shortly investigated before the section is concluded.

5.8.1 Potential benefits and disadvantages of look-ahead in AD conversion

In a typical SDM ADC, the output SNR is not limited by the quantization noise, but by the thermal noise. The reason for this balance is resulting from the fact that it is cheaper, in terms of power, to lower the quantization noise than the thermal noise. A reduction of the thermal noise can only be realized by spending more power in the analog circuits, while a reduction of the quantization noise can be realized by changing the noise-shaping filter to suppress the noise more. However, this reduction of the quantization noise comes at the cost of reducing the input signal range. Only in the situation where a signal input range as large as possible is required, and a high SNR is demanded, the quantization noise will be allowed to significantly contribute to the noise budget. In this case the thermal noise will be reduced, at the cost of spending more power, in order to reach the final SNR. In this situation it would be beneficial to realize an increase in stability of the converter, because it would enable a more aggressive noise shaping which would reduce the impact of the quantization noise on the SNR. The addition of look-ahead techniques that increase the converter stability would thus be beneficial for the design of such extreme converters. However, if the performance can be reached without look-ahead, this solution would be preferred since the cost for adding look-ahead is large, both in terms of power and silicon area.

In the situation where the transient performance or the linearity of a converter is of utmost importance, the addition of look-ahead should also be considered. However, the addition of look-ahead will come at a great power consumption and silicon area penalty. Furthermore, it should be realized that all the analog circuits should be delivering virtually ideal performance in order for the addition of look-ahead to be effective and enhance conversion quality. It is therefore questionable

if a performance increase can be realized by implementing look-ahead, especially considering that it will be difficult to realize all the steps of look-ahead algorithm in analog circuitry.

5.8.2 Feasibility of a look-ahead ADC

While in literature there are several publications that deal with the realization of a digital look-ahead SDM, no publications can be found that deal with the realization, or proposal, of a look-ahead enabled ADC. This might be explainable from the discussion above, in which it was concluded that there are no major advantages expected from the realization of a look-ahead enabled ADC. However, there could also be another reason, namely that it is difficult or impossible to realize one. In order to get more insight in the feasibility of realizing such an ADC, the potential issues for realizing the different algorithmic steps are briefly examined.

Obtaining the future input signal

From the previous sections it is clear that look-ahead can be realized with either the actual future input signal or an approximation of the future input in the case of a low-pass SDM. Since in an ADC the input signal is in analog form, there is a clear advantage of not having to store and delay the input signal and use an approximation. Still, for obtaining the best quality of look-ahead it would be beneficial to use the actual signal. In this case a discrete time delay will have to realized. Realizing the delay by means of a delay line will require re-sampling the input signal n-times, reducing the SNR of the input signal. By sampling the new input to a different capacitor instead of to the start of the delay line, this problem can possibly be reduced. There are however several disadvantages to this approach as well. First, every capacitor will be connected to the input with a different switch, an analog circuit which is very susceptible to mismatch. Presence of analog mismatches between the switches will cause variations in the sampled values, strongly reducing the maximum possible conversion quality. Next, every clock cycle the most recent sample is present in a different capacitor. It is thus not possible to make a fixed connection from a specific delay element to for example the filter input. This will require the insertion of a switch matrix between the capacitors and the filters, which is again a source of errors.

Calculation of solutions

A further big challenge for realizing a look-ahead ADC is in the evaluation of the quality of the possible future symbol sequences. In order to realize a look-ahead depth of N, the response to all 2^N possible feed-back patterns will have to be calculated. In order to have this result as fast as possible these calculations will have to be performed in parallel. However, for realizing 2^N parallel calculations, 2^N filters are required. These filters are all required to behave identical to the main SDM filter in order to enable comparison of the results. In other words, a high degree of matching between 2^N analog filters is required. Realization of a high number of matching circuits, by design or by means of calibration, is typically considered a big challenge. Furthermore, 2^N filters will require a large area and will increase the power consumption significantly.

Furthermore, the time available for calculating the filter response to the feed-back pattern is limited to the sampling period, i.e. within the sampling period the decision on the actual feed-back value has to be taken. If each of the parallel filters is set up to evaluate the effect of the complete feed-back pattern of length N, it is required to internally apply N symbols and calculate the N outputs. In order to realize this, each filter is required to run at a clock frequency which is at least N times higher than nominal. The remainder of the clock period should be used for selecting the best solution. Running filters at an N times higher rate is considered a serious challenge, certainly without increasing the power consumption by a large amount.

Still, if running a filter at a much higher speed is possible, i.e. at a clock speed which is at least $N \cdot 2^N$ times higher than the nominal sampling speed, there is no need for parallel filters and one and the same filter can be used for evaluating the 2^N feed-back responses. This approach would solve the problem of building identical analog circuits, but seems very unrealistic since already for very low values of N the clock frequency would go up with a factor of more than a hundred.

Alternatively, a setup can be envisaged where each filter only evaluates the effect of a single feed-back symbol. In this situation, at the start of the conversion, each filters internal state is loaded with results from the previous clock cycle. More specifically, in the case of a 1-bit converter, half of the solutions calculated in the previous clock cycle can be reused. Which half depends on what output symbol was selected. Although the internal clock speed of this solution is lower than in the previous case, the interaction between the filters in combination with the required precise transfer of analog quantities adds another dimension of complexity.

As a fourth alternative it is possible to pre-compute the filter response for all 2^N feed-back sequences, thus without signal present, and store these results. During signal conversion it is only required to calculate the filter response resulting from the N future input samples without feed-back. The combined input with feed-back response can now be found by applying the superposition principle, i.e. by adding the 2^N feed-back only responses to the response resulting from the input signal. The best sequence can now be selected, and the first symbol of this feed-back response is selected as the feed-back symbol. There are two main problems with this approach: calculation of the N response values resulting from the input signal, and storing of the pre-computed feed-back responses.

Calculation of the cost

Independent of what approach is taken for calculating the 2^N solutions, in every situation a cost function needs to be applied to the filter outputs. A typical cost function equals the sum of the squares of the filter output. Calculation of the square of an analog signal is far from trivial. Especially considering the fact that this calculation has to be performed in an identical fashion $N \cdot 2^N$ times. Again, extreme matching of analog circuits is required.

Selection of the best solution

Once the 2^N cost values have been calculated, they need to be compared in order to find the solution with the lowest cost. Since the cost difference between solutions could be small, this comparison is required to be precise. A large number (approximately $N * N/2$) of comparators with small input referred offset is therefore required. A small amount of digital logic processing the comparison results will finally be able to deliver the optimal feed-back symbol.

5.8.3 Hybrid look-ahead ADC

From the discussion above it is clear that realizing look-ahead in the analog domain will be a challenging task. An alternative approach for realizing an ultra high quality ADC with a 1-bit output should therefore be considered. Instead of converting the analog signal directly to a 1-bit digital signal, a hybrid approach consisting of a multi-bit SDM ADC and a DD converter with look-ahead can be envisaged.

In this scenario the SDM ADC should preferably oversample the input signal at a rate much higher than the final desired oversampling ratio, e.g. 256 or 512 times oversampled instead of 64. The higher oversampling rate will enable a good transient response, and large signal band. The combination of the high oversampling with a multi-bit quantizer, e.g. a 5-bit quantizer, will enable a large input range, a high SNR, and avoid the typical 1-bit SDM problems. The result of this AD operation is thus a very high quality signal. The only problem with this signal is that it is not in the desired format. In order to realize a high quality conversion from multi-bit to 1-bit and change the signal to the correct sampling rate, a DD SDM converter with look-ahead can now be used. This setup is depicted in fig. 5.13.

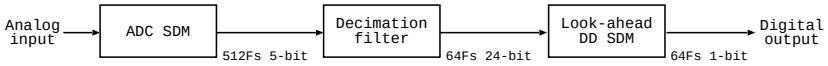


Figure 5.13: Example hybrid look-ahead ADC, consisting of a 512 times oversampled 5-bit output ADC, digital decimation filter that delivers a 64 times oversampled signal with 24-bit resolution, and a 1-bit digital-to-digital look-ahead SDM.

Before the ADC output can be applied to the DD SDM, it will first need to be low-pass filtered and decimated. In order to keep the highest quality, especially for the transient response, the low-pass filter should have a very slow fall-off which only starts at a high frequency. For example, although the final output will have a signal band of 20 kHz, it is beneficial for the transients if the low-pass decimation filter has a much higher corner frequency, e.g. 80-100 kHz. The disadvantage of having a high corner frequency is that more of the AD SDM generated noise will be passed to the DD converter. Especially if the signal band of the ADC is relatively narrow, i.e. not much wider than the final desired signal band, the out of band noise of the ADC might become problematic. More specifically, the DD SDM will encode the ADC generated noise and add its own quantization noise, resulting in a higher out of band noise floor. The solution to relax this problem it to design the ADC to have a large noise free signal band.

The output of the decimation process, a multi-bit signal at the output rate, will be finally passed to a DD converter with look-ahead. The DD converter will transform the multi-bit signal to a noise-shaped 1-bit signal. The final result is a combination of the best of two worlds; analog-to-digital conversion using a multi-bit SDM ADC and multi-bit

to 1-bit conversion using a look-ahead enabled DD SDM.

5.8.4 Conclusion

It is not clear what benefits look-ahead can bring to 1-bit Sigma-Delta based analog-to-digital conversion, but what is clear is that the implementation of the required algorithm steps in the analog domain will be very challenging. An alternative approach for indirectly realizing a higher quality 1-bit Sigma-Delta ADC has been identified, namely a hybrid approach. In the hybrid approach, analog-to-digital conversion is performed with a high speed multi-bit SDM ADC, which can realize a higher quality analog-to-digital conversion than a 1-bit SDM ADC. In the digital domain the multi-bit SDM output is subsequently converted to a 1-bit format using a look-ahead enabled DD SDM, resulting in a high quality bitstream.

5.9 Look-ahead DD conversion

In an analog-to-digital converter thermal noise and analog circuit imperfections are typically the limiting factors for signal quality. In a digital SDM, however, all the signals are of a digital nature and, in principle, thermal noise should not influence the outcome of the signal conversion. Furthermore, as long as the circuit imperfections do not alter the digital circuit functionality, the outcome of the digital operations is unchanged and the imperfections are non-existent from a signal perspective. In contrast to analog circuits, digital circuits show perfect reproducibility without any variation from realization to realization and will always deliver the same outcome when presented with the same stimuli. On the other hand, more or less comparable to the influence of noise on the dynamic range of an ADC, is the effect of a too small word width in a DD converter. If not enough bits are assigned to store the results of signal operations, over- or underflow will occur. In the case of underflow noise will be added to the conversion result, reducing the dynamic range. In the case of overflow an unpredictable output will result. A better solution is therefore to clip the signal to the maximum possible value, what will result in distortion which is preferred over an undefined signal. This behavior is comparable to the clipping to the supply levels in an ADC. Because the circuit is all digital, in practice errors of this type can be easily avoided. By performing extensive behavioral circuit simulations, the required word widths can be determined with a high degree of certainty, despite the non-linear quantizer transfer. At the circuit level

design, in a custom hardware realization it should also be guaranteed that the hardware can be clocked at the desired clock frequency without introducing errors, for example caused by violating memory setup requirements. If these basic constraints are all fulfilled, it can be argued that a digital SDM will perform ideal signal conversion.

Although it is possible to realize an ideal DD Sigma-Delta converter that does not add noise or distortion because of circuit imperfections, the conversion process will still typically cause a reduction of the signal quality. This reduction is caused by the addition of quantization noise, and possibly by distortion introduced by the noise-shaping process. The amount of quantization noise that is added in the signal band depends on the noise-shaping characteristics of the converter, i.e. the loop-filter order, loop-filter corner frequency, the oversampling ratio, and the number of bits of the quantizer. The quantizer is the main cause of the generation of distortion components, especially in the case of a 1-bit or few-bit quantizer. When a low order loop filter is used, little de-correlation of the quantization noise is achieved, and little suppression of the quantizer generated distortion is resulting.

In order to reduce the distortion tones, an often applied solution is to dither the digital SDM. By dithering the SDM, i.e. adding an approximation of a noisy signal which is typically generated with a pseudo random generator to the quantizer input, the decision of the quantizer is changed in a random fashion when it is close to a decision threshold. As a result, the periodic patterns which are required for generating tones are disturbed, and the distortion tones are attenuated. By adding a larger amount of dither a stronger reduction of tones is realized. However, the addition of dither is not without penalty: the output SNR is degraded and the stable input range is reduced. Instead of dithering the quantizer strongly, it is possible to use a small amount of dither in combination with a high order loop filter that provides more de-correlation of the quantization noise. However, also in this case the highly non-linear transfer of a 1-bit quantizer is often the cause of harmonic distortion components in the output, especially for large input signals. Furthermore, the stable input range is reduced because of the high order loop filter.

Instead of dithering the modulator, the application of look-ahead techniques should be able to provide an alternative solution for the suppression of distortion. Simultaneously, the look-ahead algorithm will increase the stability of the converter, allowing for a larger input range. Thus, instead of suppressing distortion at the cost of a reduced input range, an increased input range with less distortion should be realizable

by adding look-ahead functionality. However, as reasoned in sec. 5.7, a significant amount of look-ahead might be required in order to detect and suppress low-frequency distortion. If only a small amount of look-ahead is applied, an increase in stability is already expected, but no significant reduction of distortion is foreseen. Therefore, as an alternative it should be possible to apply a significant amount of dither for reducing the distortion, and rely on the look-ahead algorithm for maintaining the same stable input range. Another alternative that could possibly improve the conversion quality is to apply look-ahead while increasing the filter order in combination with a small amount of dither.

A further advantage of look-ahead, besides improving the linearity and increasing the stability, is the realization of a better transient response as already mentioned earlier. In stereo music recordings a good quality transient response is very important, since the human brain performs spatial localization by analyzing phase differences. An accurate transient response will thus result in a higher quality recording. Application of look-ahead for realizing the highest quality SA-CD bitstreams seems therefore beneficial.

Besides the already mentioned disadvantages of dithering, there is another major disadvantage associated with dithering if the bitstream is intended for SA-CD usage. By adding dither, i.e. adding randomness, the entropy of the signal is increased. Because of this increased entropy the lossless data compression that is used by SA-CD will become less efficient and more disc space will be required to store the result after compression. As a result, the potential disc playback time will reduce and mastering issues could be resulting. Especially first generation studio-grade digital Sigma-Delta Modulators, which offered a very high audio quality, suffered severely from this problem and caused manufacturing issues on numerous occasions. As a work-around procedure, the high quality audio signal was typically re-quantized with a lower quality modulator such that the entropy in the signal was reduced, and a higher compression ratio was achieved. Second generation DD converters were, typically, fifth order Sigma-Delta Modulators instead of sixth or seventh order, and applied a minimal amount of dither such that distortion was reduced to an acceptable level and that the compression gain was higher. Still, also with these modulators the compression gain is on the lower limit, and a higher compression gain in combination with ultra high audio quality would be appreciated.

For solving the potential SA-CD playback time issue a look-ahead based solution can be envisaged that provides both a very high quality audio conversion and a good compression ratio. However, for this to work, it is

key to not rely on dither for obtaining good linearity since this will have a negative impact on the compression ratio. As a consequence, a large look-ahead depth will be required. Since a straightforward implementation of the look-ahead approach will not be able to efficiently provide large amounts of look-ahead because of the computational complexity, realization of a computationally efficient look-ahead algorithm is key for implementing a DD look-ahead SDM that is suitable for SA-CD mastering. In ch. 6 the possibilities for efficient look-ahead are investigated.

5.10 Conclusions

From the generic noise-shaping quantizer of ch. 4 a noise-shaping quantizer model with the possibility of performing look-ahead has been derived. The look-ahead capability is realized by providing the cost function with information about the available output levels. In combination with the future input signal values it is now in principle possible to look ahead in time.

In practical realizations of the full look-ahead algorithm, the cost of all the possible feed-back patterns for the next clock cycles is calculated, the best solution is selected, and the first symbol of this solution is used in the main conversion as the feed-back signal. In real-time systems, calculation of the future responses is realized by storing the input signal in a delay line and working on delayed versions of the signal.

A linear model of a generic look-ahead SDM has been derived. The predicted NTF of a feed-forward and feed-back look-ahead modulator are equal when the same filter coefficients are used, as is the case for a normal SDM. Comparison of the NTF with that of a normal SDM reveals that, according to the linear model, the look-ahead SDM will achieve a slightly lower SNR if the same filter coefficients are used. The STF of a look-ahead SDM is similar to that of a normal SDM, but also here subtle differences exist. Most interesting, the predicted STF of a feed-forward look-ahead SDM is equal to a unity gain transfer without phase shift.

Potential advantages of the look-ahead principle are an increase in the stability of the converter, an improvement of the linearity, and a better match between the input signal and the output signal, reflecting e.g. in a better transient response. The disadvantage of applying look-ahead is a severe penalty in the power consumption and the implementation cost. A potential disadvantage of a look-ahead SDM, compared to a normal SDM with the same loop filter, is the lower SNR. However, this effect

can be compensated by using a slightly more aggressive loop filter, as demonstrated in chapter 7.

The feasibility of a look-ahead enabled ADC is estimated as very low. Considering the fact that the thermal noise and the analog circuit imperfections are the main contributors to a reduced signal conversion quality, the potential performance gain is also low. In a digital-to-digital SDM, look-ahead technology can improve signal conversion quality, especially in SA-CD mastering applications where signal quality is of utmost importance. Furthermore, by increasing the look-ahead depth to a very large value, it is expected that the distortion generated by the 1-bit quantizer can be suppressed without the use of dither. As a result, the entropy of the signal will reduce and a higher lossless compression gain, i.e. a longer playback time, will be realized. However, for the look-ahead approach to be practically feasible in a digital-to-digital converter, the computational complexity associated with the look-ahead depth should be reduced.

Chapter 6

Reducing the computational complexity of look-ahead DD conversion

In the previous chapter it was concluded that look-ahead modulation may bring great improvements in conversion quality, but that an improvement in computational efficiency is required for the approach to be of practical interest. Two different possibilities for realizing a higher efficiency are explored. In sec. 6.1 the possibilities for reducing the number of computations required for full look-ahead modulation are investigated. In sec. 6.2 an alternative to full look-ahead modulation is introduced, called pruned look-ahead modulation. A higher efficiency is now realized by investigating only a subset of the complete solution space. Since the latter approach has more potential, a number of realizations to verify the theory are proposed in sec. 6.3. The chapter is concluded in sec. 6.4.

6.1 Full look-ahead

In the full look-ahead algorithm (sec. 5.5) all the possibilities for extending the running bitstream with N symbols are investigated. This means that for a 1-bit converter there are 2^N sequences with a length N to evaluate in order to extend the bitstream with a single symbol. A straight-forward implementation of the algorithm could use a double-nested loop to calculate all the filter outputs, one after the other. The computational load of this approach is comparable to the calculation of $N \cdot 2^N$ outputs of a normal SDM. A look-ahead of 10 symbols would thus

require the calculation of 10 240 loop-filter outputs, i.e. without the additional overhead of selecting the best solution the workload is 4 orders of magnitude larger than that of a normal SDM. A more efficient computation of the solution is clearly desired and can be realized in several ways.

6.1.1 Complete response calculation with reuse of intermediate results

The number of filter outputs to evaluate can be reduced by reusing intermediate results and avoiding the unnecessary re-computation of already generated results. From the 2^N bitstreams to evaluate, half of these start with a '0' and half with a '1'. In the straight-forward solution the response to this first symbol is calculated 2^N times, i.e. 2^{N-1} times for the '0' symbol and 2^{N-1} times for the '1' symbol. Since the initial condition is identical for all these runs the outcome will also be identical, and the amount of evaluations can be reduced to two. Each of the two results will be used 2^{N-1} times in the total evaluation.

For the second symbol, four evaluations are required, i.e. a '0' and a '1' will be appended to the first symbol which is either '0' or '1'. The results will each be re-used 2^{N-2} times for the evaluation of the remaining $N-2$ bits.

By applying full re-use of results, the total amount of filter output evaluations can be reduced to $2 + 4 + 8 + ... + 2^N = 2^{N+1} - 2$. For a look-ahead depth of 10 the number of filter output calculations equals 2046, approximately 20% of what is needed in the brute-force approach. The disadvantage of this approach is that the intermediate filter states need to be stored in memory. The number of required memory locations equals $1 + 2 + 4 + ... + 2^{N-1} = 2^N - 1$.

6.1.2 Select and continue with half of the solutions

Another scheme for reusing results in order to reduce the number of computations is possible. Once the 2^N possibilities have been calculated, the best solution is selected and the first symbol of this solution is used as feed-back value. In the next clock cycle, again 2^N possibilities for continuing the bitstream are calculated. However, the first $N-1$ bits from these 2^N solutions have already been evaluated in the previous clock cycle, i.e. the last $N-1$ symbols of the solutions that started with the same symbol as the one used for the feed-back are now the first

$N-1$ symbols. Re-computation of these results is unnecessary and can be avoided. If the feed-back symbol is a '1', all the solutions that start with a '1' are kept, and vice versa. The required calculations are limited to those necessary for adding a last symbol to the already pre-computed 2^{N-1} solutions. This means that only the impact of 2^N single symbols needs to be evaluated instead of the impact of $N \cdot 2^N$ symbols.

This approach reduces the number of evaluations to a fraction of $\frac{1}{N}$ of the number required in the brute-force approach. In the specific case of a look-ahead depth of 10, the computation of only 1024 filter outputs is required instead of the 10 240 required by the brute-force approach, i.e. a reduction of 90%. The disadvantage of this solution is that 2^N memory locations to store the filter states are required, and that at every clock cycle the memory locations need to be selectively updated.

6.1.3 Linear decomposition of the filter response

The number of filter output evaluations can be further reduced compared to the situation where the intermediate results are reused. Key is the observation that although the SDM loop is non-linear, the transfer function from the input of the filter to the output of the filter is linear. Therefore it is possible to calculate the filter output in two passes, and afterwards sum the results to obtain the final result. In the first step no feed-back signal is applied to the filter and only the response resulting from the input signal in combination with the filter state is calculated and stored. In the second step the filter state of the converter is reset to zero, and only the feed-back signal is applied. If the two filter output signals are added, the same signal is obtained as in the normal approach where the response is calculated in one pass.

This split in computations is beneficial because in principle now only the response to the signal input needs to be calculated in real-time. The response to the feed-back signal is independent of the filter state or input signal and can be pre-computed and stored in a memory. Thus, at the expense of a memory that holds the 2^N different filter responses with length N, it is possible to obtain the combined filter response by simply adding the response caused by the input signal and the pre-computed responses. Without adding complexity, the amount of memory can be halved by only storing all the responses that start with a '1' symbol. The response of sequences that start with a '0' symbol can be obtained by subtracting the pre-computed response of the complementary sequence. For example, the response to the sequence "01011" can be obtained by subtracting the pre-computed response to "10100".

Since the calculation of the filter response is a completely linear operation, the amount of storage memory can be reduced to only 1 filter response, at the cost of additional summation operations. In this scheme the filter impulse response is pre-calculated and stored. From the impulse response the feed-back responses can be generated by adding or subtracting a delayed and shortened version of the impulse response. The number of additions required to calculate a response of length N equals $1 + 2 + ... + (N-2) + (N-1) = N \cdot (N-1)/2$. This results in 45 additions to calculate the filter response to a feed-back sequence of length 10 and another 10 additions to add the result to the input signal response.

Instead of generating all 2^N responses in this way, reuse of already generated results is possible again, which will reduce the number of filter outputs to generate drastically. If only the solutions that start with a '1' symbol are calculated, there are $\frac{2^{N+1}-2}{2} = 2^N - 1$ filter output evaluations required (from sec. 6.1.1). The number of additions required to calculate these filter outputs by adding impulse responses equals $1 \cdot 0 + 2 \cdot 1 + 4 \cdot 2 + 8 \cdot 3 + ... + 2^{N-1} \cdot (N-1) = \sum_{n=0}^{N-1} 2^n \cdot n$. For a look-head depth of 10 this results in 8194 additions. Another $2^{N+1} - 2 = 2046$ additions are required for summing the feed-back responses and the response from the input signal, resulting in a total of 10 240 additions.

For comparison, in the original brute-force approach there are 10 240 calculations of loop-filter output values required. If a fifth order loop filter is assumed, there are 5 additions required for realizing the integrators. Another 5 additions are required to sum the feedback signals, and at least 4 multiplications are required for implementing the filter coefficients (from the original 5 coefficients 1 can be made equal to unity in some cases). If resonator sections are required the number of additions and multiplications is even higher. By selecting conveniently chosen coefficients, the multiplications can be replaced by shift-and-add combinations. Assume each multiplications therefore takes 2 additions. The total number of additions required to calculate one filter output then comes to 18. In order to calculate the 10 240 loop-filter output values in the brute-force approach, approximately $184.5 \cdot 10^3$ additions are required. By decomposing the filter response calculation this number is reduced to 5.56%, which equals approximately 570 filter output evaluations.

6.1.4 Conditional computation of the solutions

Instead of calculating all 2^N responses completely for every time-step, it is also possible to evaluate a varying number of, possibly partial, responses and still obtain the correct outcome. The algorithm is known as the stack algorithm [36]. A related algorithm that uses heuristics to speed up the calculations, but that cannot guarantee to find the optimal result, is the Fano algorithm [15].

In the stack algorithm, initially the cost for the two possibilities for the first symbol is calculated (fig. 6.1 a)). The symbol that results in the lowest cost is selected, and a '0' and a '1' are added to this solution. At this moment there are 3 partial results calculated (fig. 6.1 b)). From all the (partial) results the one with the lowest accumulated score is selected, and again a '0' and a '1' are added (fig. 6.1 c)). This process is repeated until the look-ahead depth of N has been reached by a path. If this path has the lowest accumulated cost from all the partial results, the solution with the lowest cost has been found. If a partially completed path has a cost which is lower than the cost of the fully completed path, the search process is continued by extending the partial path with a '0' and a '1'. This situation is depicted in fig. 6.1 d). This process is continued until a path with the desired look-ahead depth has been constructed that has a lower cost than all the calculated results. At this moment it is certain that the cheapest path has been found and there is no need to calculate the results for the other paths. In the example, for the case $N = 3$, the path with the lowest cost has been found after step d).

In principle, a large saving in the amount of filter output calculations can be realized with this strategy. In practice, however, the algorithm introduces significant overhead, mainly caused by the selection process, and a reduction instead of improvement in throughput is realized [6]. Furthermore, the algorithm results in a workload which is not constant, which is problematic for real-time applications.

6.1.5 Calculating multiple output symbols per step

In the full look-ahead algorithm typically only one output symbol is generated per clock. As an alternative it is possible to output multiple symbols at a time at a lower rate. For example, if two output symbols are generated instead of one, there is twice the time available to generate those. Thus, instead of producing one symbol each clock cycle, every second clock cycle there will be two output symbols.

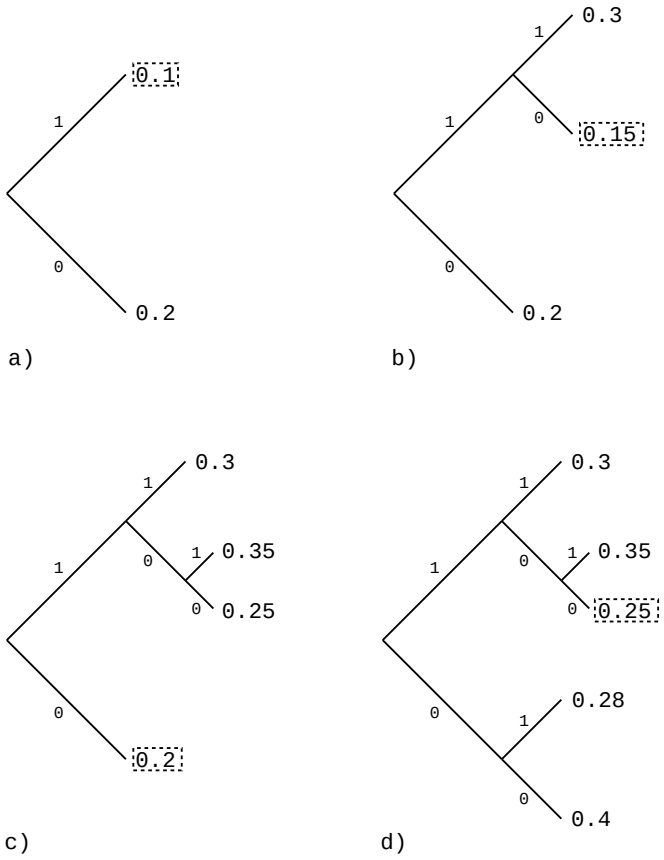


Figure 6.1: Example conditional construction of a look-ahead tree. The accumulated path score is printed at the end of each branch. The lowest score is indicated by a dashed box.

In order to obtain multiple output symbols at a time no significant change to the algorithm is required. Instead of only using the first symbol of the best feed-back sequence, now the first n symbols will be used. If, for example, two output symbols are generated per step, from the (partial) look-ahead tree of fig. 6.1 d) the output sequence '10' would be selected. As a result the amount of effective look-ahead is different for the sequence of output symbols, i.e. the first symbol is based on a look-ahead depth of N, the second symbol is based on a look-ahead depth of $N-1$, and so on. Thus, the probability of selecting the optimal

symbol reduces for every next output symbol. However, the advantage is a reduction in the computational load. For example, if two output symbols are generated, the look-ahead tree will need to be extended with two levels in two clock cycles. This results in the calculation of $2^{N-1} + 2^N = 1.5 \cdot 2^N$ values in two clock cycles, which equals $0.75 \cdot 2^N$ calculations per clock cycle, or a reduction of 25% compared to the one symbol per clock situation.

In the situation where it is not acceptable to have a reduced look-ahead depth for any of the output symbols, the approach is not favorable. For example, if at least a look-ahead depth of N is required and two symbols are generated per clock, the look-ahead depth for the first symbol will have to be increased to $N + 1$. The total amount of computations, distributed over two clock cycles, becomes $2^N + 2^{N+1} = 3 \cdot 2^N$. The load per clock cycle thus increases by 50% compared to the traditional single symbol per clock approach.

6.1.6 Summary

Compared to the straight-forward brute-force full look-ahead approach, the number of computations can be reduced by changing the algorithm for calculating the path scores, without affecting the outcome of the process.

For look-ahead depths up to approximately 20, the most efficient solution in terms of computational cost is based on a linear decomposition of the filter response evaluation. The number of filter output evaluations required by this approach equals approximately $N \cdot 2^N/18$ for a simple fifth order modulator. For a look-ahead depth of 10 this results in the equivalent of 570 filter output evaluations; for a depth of 16 the number of evaluations becomes $58 \cdot 10^3$.

For look-ahead depths of 20 and above the computationally most efficient solution is to select half of the solutions from the previous clock cycle and to continue from there. The number of filter outputs to calculate at every time step equals 2^N. Thus, already more than 1 million evaluations are required for a look-ahead depth of 20, and the number doubles with every increment of the look-ahead depth.

In comparison to the brute-force approach the computational savings can be summarized as a factor N for large look-ahead depths, while for small look-ahead depths the savings can be up to two times higher. An alternative approach based on the conditional computation of solutions by using the stack algorithm, although in theory superior to the full

calculation methods, introduces significant overhead and does therefore not result in any computational savings.

6.2 Pruned look-ahead

In a pruned look-ahead modulator only a fraction of all the possible feed-back patterns are examined. The motivation for this approach is given in sec. 6.2.1. Next, the basic pruning approach is detailed in sec. 6.2.2. In order to enable large pruned look-ahead depths at minimal computational cost, reuse of results is required which is not possible in the basic pruning approach. The solution to enable efficient pruned look-ahead and the implications of this approach are presented in sec. 6.2.3. Finally, the insights obtained in this section are summarized in sec. 6.2.4.

6.2.1 Motivation for pruning

In the case of a full 1-bit look-ahead modulator all the 2^N possible solutions are investigated. As a result, despite algorithmic optimizations, for large look-ahead depths the computational load doubles with every increment of the look-ahead depth. The stack algorithm tries to avoid the calculation of the complete set of 2^N solutions, and will typically require less evaluations to find the cheapest solution. Unfortunately, the algorithm introduces a lot of overhead, causing an overall decrease in efficiency instead of a speedup. Still, it is reasonable to assume that the evaluation of only a subset of the complete solution space can in typical cases increase the efficiency of the conversion if implemented properly.

In a pruned look-ahead modulator, instead of calculating all the 2^N possible solutions, only a subset of the solution space is evaluated. With a proper selection criterion of what solutions to investigate, the outcome of this approach can be close or possibly even identical to the outcome found when all solutions are evaluated. If more solutions are investigated, i.e. less pruning, the likelihood of finding the optimal solution will increase at the cost of a reduced computational gain. Thus, by varying the amount of pruning a tradeoff between the computational load and the conversion quality can be realized. If the selection heuristic is of a high quality, it is expected that with a small number of solutions, i.e. a high level of pruning, a close to optimal solution can still be found. As a result it should become feasible to realize a larger look-ahead depth and to realize an overall higher conversion quality.

In the full look-ahead algorithm, especially when the look-ahead depth is large, a significant amount of unnecessary filter output evaluations are typically performed. By studying a simple but representative input signal this phenomenon can be understood, and it will become clear that pruning can enable a higher computational efficiency.

For example, consider the situation of an interpolative SDM with the input signal equal to a large DC value. In the case of a traditional SDM the output signal will have the same average value as the input signal. This holds for the local average and also for the global average. In the case of a look-ahead SDM these rules still apply, and the short term average of the output will therefore have to match the DC input value. If we consider a 1-bit SDM, it is clear that the output pulse density will have to match the DC level. For example, if the input DC level equals $FullScale/2$, the output is required to have, on average, three ones and one zero symbol per four bits $((1 + 1 + 1 - 1)/4 = 2/4 = 0.5)$. The number of possibilities, even without taking into account the additional constraints imposed by the noise shaping, to generate such a sequence are limited. If we consider a relatively short sequence of length 12, there are only 220 out of the 4096 possible binary sequences that result in the proper average, i.e. only 5.4% of the sequences could potentially show a good noise-shaping characteristic and match the input signal. However, in the full look-ahead algorithm every bit combination of length N is evaluated, also the ones that are not resembling the input signal at all.

If it would be possible to detect a-priori which paths do not show a good match with the input signal, the computations for these solutions could be spared, and an identical conversion result could be realized at a lower cost. Fortunately, in the case of a large look-ahead depth these paths can be detected with reasonable certainty at a limited processing cost, enabling pruned look-ahead modulation.

6.2.2 Basic pruned look-ahead modulation

For the efficient realization of a large look-ahead depth reuse of intermediate results is essential. From sec. 6.1 it is known that the most efficient approach is to reuse half of the solutions from the previous time step (see sec. 6.1.2). However, in the case of pruned look-ahead modulation it is not possible to apply this approach in its original form, as will be demonstrated later in this section. Therefore, first the basic approach of pruned look-ahead will be investigated, in which the look-ahead tree is fully constructed every clock cycle.

Consider the case of a full look-ahead modulator in which the look-ahead

tree is built with reuse of intermediate results (sec. 6.1.1) and the tree is constructed level by level, i.e. all the solutions up to depth n will be generated before the tree is extended to depth $n + 1$. As a result of this approach, it is possible to see all the path scores gradually develop in parallel. Initially, all paths will have approximately the same score, but when the depth of the tree is increasing the path scores will start to show variations as a function of the quality of their match with the input signal.

Imagine the look-ahead tree is fully completed for a depth n. It can be understood that a path that does not match the input over these n time steps will not be able to show a good match over $n + 1$ time steps. Thus, the score which was readily accumulated can be considered as an indication of the quality of the new path. Especially when the look-ahead depth is large, the difference between the score of a path that matches the input and a non-matching path will be large since the score will have developed over a number of clock cycles. In this situation it is therefore possible to detect which solutions have a chance of becoming the best solution, and which solutions do not require evaluation because they have accumulated a higher score.

However, if a solution is not expanded at depth n, the look-ahead tree will be incomplete from this point onwards and the number of solutions that can be investigated for a depth of $n + 2$ by simply expanding the look-ahead tree of depth $n + 1$ will have been reduced. It is therefore crucial to not reject too many paths since the solution space could be reduced too severely, resulting in a lower quality conversion. On the other hand, when not enough solutions are removed the number of required computations could grow exponentially, and no savings compared to the full look-ahead approach are realized.

A solution to the problem described above is the following. Instead of removing a varying number of paths at every time step, initially when the look-ahead tree is constructed no paths are removed, and only once the number of solutions is large enough a fraction of the paths will be removed. The fraction that is being removed should depend on the rate of the growth of the solution space, and should be such that the number of paths stays constant.

For example, if the number of paths to investigate is limited to 1024, a full look-ahead tree of depth 10 can be calculated for a 1-bit quantizer. If the look-ahead depth is increased to 11 while at the same time the number of computations is limited to 1024 evaluations, only half of the solutions for the depth of 11 can be calculated. This is realized by selecting the 'best' half of the solutions at depth 10 and by expanding

only those solutions. After this operation again 1024 solutions exist, but a pruned look-ahead depth of 11 has been realized. If from this current set of 1024 solutions again half of the solutions are selected, a pruned look-ahead depth of 12 can be realized. From the 4096 (2^{12}) possible bitstreams with length 12 only 1024 possibilities will have been fully investigated.

A graphical demonstration of the process of building a pruned look-ahead tree is shown in fig. 6.2. For simplicity, the number of parallel paths is limited to 4 in the example. The intermediate steps required to obtain a pruned look-ahead depth of 4 are separately shown (figs. a..c). From the possible 16 sequences only 4 are fully examined. The best path is indicated in bold, the solutions which are not investigated are dashed.

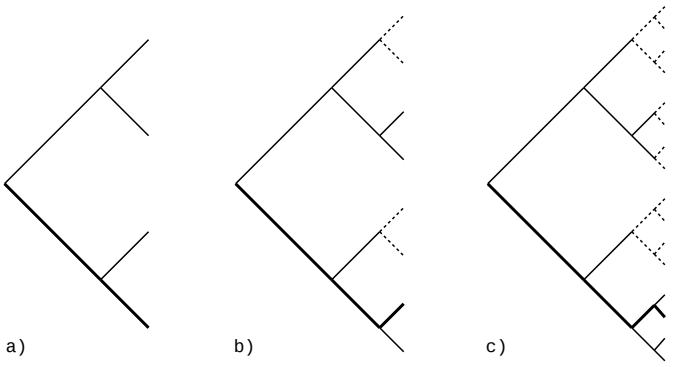


Figure 6.2: Example construction of a pruned look-ahead depth of 4 with 4 parallel paths (figs. a..c). The best path is indicated in bold, the solutions which are not investigated are dashed. Starting at a look-ahead depth of 4 (fig. c) the 4 best paths agree on the output symbol.

Once the desired look-ahead depth has been realized, the output symbol is determined and time is advanced. The procedure is identical to that of the full look-ahead algorithm, i.e. it is determined which path has the lowest cost, and the first symbol of this path is selected as the output symbol and used as feed-back value for the modulator. With the feed-back symbol the main modulator can be updated and the next input sample can be processed. Reuse of part of the pruned look-ahead tree is not easily possible because of the varying number of solutions that are maintained. Therefore, the complete pruned look-ahead tree has to be built again from scratch, limiting the computational savings compared to full look-ahead.

Key for the correct functioning of this approach is the selection criterion that is applied to select the 'best' solutions (see 5.3.1). With a proper criterion the most promising solutions are kept while building the look-ahead tree, and the solutions that are less likely to become the best match with the input signal are discarded. In this case the outcome of the pruned conversion will be equal to what would be realized with a full look-ahead conversion, except that less computations are required. If the selection heuristic is of a low quality the look-ahead tree will possibly not contain all relevant potential solutions and a reduced conversion quality could be resulting.

6.2.3 Pruned look-ahead modulation with reuse of results

The scheme described in the previous section can be used to generate a pruned look-ahead tree from scratch. However, a problem exists when time is advanced and re-use of computations is desired. In the full look-ahead algorithm, the best path, i.e. the path with the lowest cost, would first be selected. Next, from this selected path the output symbol is determined, and all the solutions that start with the output symbol are selected and passed on to the next clock cycle. Because half of the solutions start with a '0' symbol and half with a '1' symbol the number of solutions that remains after the selection is always the same. In the pruned case in principle the same procedure could be applied for selecting the output symbol, but a problem exists when removing the solutions that do not start with the selected output symbol. More specifically, it cannot be guaranteed that half of the solutions remain once all paths that do not start with the selected output symbol are removed. As a result, the number of solutions still under investigation in the next time step could be reduced or increased.

To illustrate this problem, consider the example pruned look-ahead tree of fig. 6.2, where with 4 parallel paths a pruned look-ahead depth of 4 is realized (final result in fig. c). Independent of which of the 4 paths is the best choice, the output symbol will be the same. Therefore, the number of solutions that remain after removing the solutions that do not start with the correct symbol is still 4, i.e. no solutions are removed. If the example pruned look-ahead tree of fig. 6.3 is considered, only half of the solutions will remain after the paths that start with a different symbol than the best path have been removed. Thus, depending on the status of the pruned look-ahead tree a varying number of paths will remain. In the example only 4 parallel paths are considered, and either half or all

paths remain, but in the generic case the number of paths that remain can vary between 2 and N in multiples of 2.

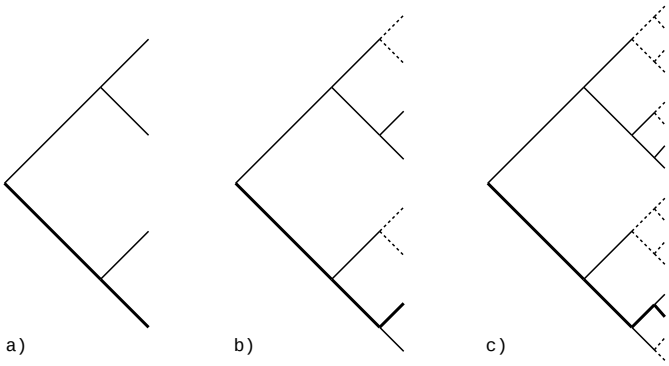


Figure 6.3: Second example construction of a pruned look-ahead depth of 4 with 4 parallel paths (figs. a..c). The best path is indicated in bold, the solutions which are not investigated are dashed. The 4 best paths do not agree on the output symbol for a look-ahead depth of 4.

When pruning a look-ahead tree there is a combination of two independent mechanisms that influence the amount of solutions that remain in the look-ahead tree. On the one hand there is the procedure for determining the solutions that are promising and that will be continued. On the other hand there is the output symbol selection that influences the amount of solutions that remain in the look-ahead tree. If no re-use of results from the previous conversion is required, the output selection mechanism does not exist and the problem of a varying number of solutions is solved. However, in such a solution only a limited look-ahead depth can be efficiently realized. For an efficient realization of large look-ahead depths re-use of results is required, and a combined control over the two mechanisms is needed such that the number of solutions under investigation remains (near) constant. However, since both mechanisms are independent, such a combined control is difficult to realize in the general case. For the specific case of a very large (pruned) look-ahead depth an elegant solution exists.

Consider the situation of a very large pruned look-ahead depth, for example 100, with the number of evaluations per time step limited to 1024. Let the look-ahead tree be extended 1 level per conversion cycle. Initially, during the first 10 clock cycles, no pruning is required and the number of solutions in the tree will double every clock cycle until a

depth of 10 is reached. Now the tree will be extended to depth 11 while maintaining 1024 solutions. This is realized by selecting the most promising 512 solutions and extending those, such that again 1024 unique solutions exist. This process is repeated until a pruned depth of 100 is realized. Since the target look-ahead depth has now been reached, the first output symbol will need to be determined. The output symbol can be found by selecting the path with the lowest cost, and by taking the first symbol of this sequence. On the other hand, if the second best path would be selected, typically the same symbol would be found. In fact, if the two paths would be compared, only minimal differences between the sequences would be found, and those differences would be mainly concentrated in the last part of the sequences. The explanation of this phenomenon is as follows.

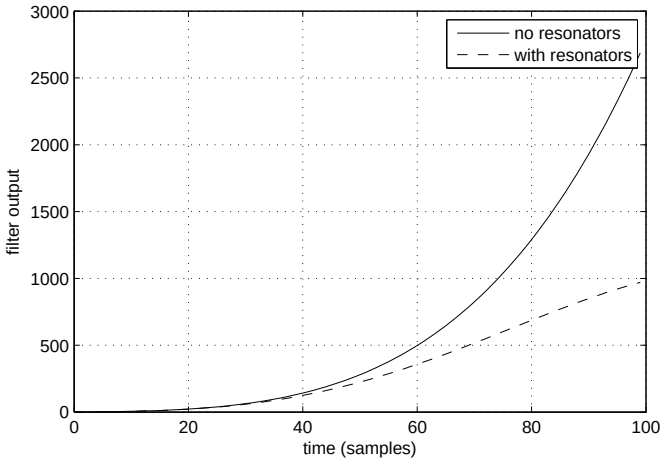


Figure 6.4: First 100 samples of the impulse response of a 5th order loop filter with a corner frequency of 100 kHz, once with and once without resonator sections (12 kHz and 20 kHz).

If we consider a high order modulator with a loop filter without resonators, the impulse response of the loop filter only (without the quantized feed-back) is constantly increasing over time. If the loop filter contains resonator sections the impulse response has a slightly different shape, i.e. in this case only the envelope is increasing over time, but this has no significant impact, as will be discussed later. As an example, the first 100 values of the impulse response of a fifth order filter with a corner frequency of 100 kHz is shown in fig. 6.4, once with and once without res-

onator sections. Because of the constant (envelope) growth, the longer ago an impulse has been passed to the loop filter, the larger the loop-filter output becomes. Since the loop filter is a linear system, the combined output for a series of input values equals the sum of all the individual filter responses. As a result, under the assumption of a constant input signal, the first symbol of a trial feed-back sequence has a larger impact on the combined loop-filter output than the second symbol, which again has a larger impact on the output than the third symbol, etc. Thus, it is critically more important for older feed-back symbols to match the input signal than for more recent feed-back symbols, since the difference between the input signal and the feed-back symbol is weighed stronger, i.e. the difference is multiplied with the impulse response. Therefore, the iterative selection operation that is applied while building the pruned look-ahead tree will over time gradually select and keep those solutions that have the best combined match for the first series of symbols. Once the look-ahead tree has reached a large enough depth, the first symbol of all the solutions that are maintained will be equal, and variations will only occur at later time instants. This situation where the first series of symbols is equal for all the solutions is schematically depicted in fig. 6.5.

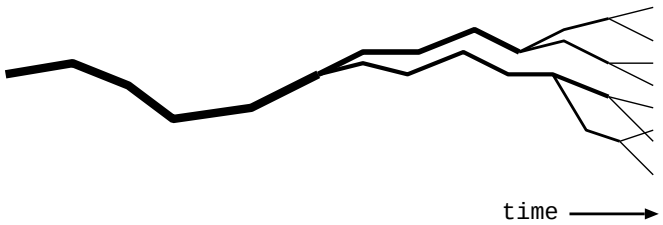


Figure 6.5: Schematic representation of a gradual convergence of all the pruned solutions. Thickness of the lines is proportional with the number of equal solutions.

If it is now assumed that in the example situation the first symbol of all the 1024 sequences is identical when the look-ahead tree has reached a depth of 100, the procedure for selecting the output symbol has been reduced to simply selecting the first symbol of any of the solutions. Note that this output symbol is resulting from the input that was applied 100 conversion cycles ago. Furthermore, since all sequences start with the same symbol, the number of valid sequences after selection of the output symbol will be identical to the number before selection, i.e. no solutions will be removed because of a non-matching symbol. In the next conversion step the extension of the look-ahead tree from a length

of 99 to a length of 100, while maintaining 1024 solutions, can now be realized by simply selecting the best 512 solutions and expanding those. Thus, pruned look-ahead modulation with re-use of the results from the previous conversion cycle is possible with minimal computational overhead.

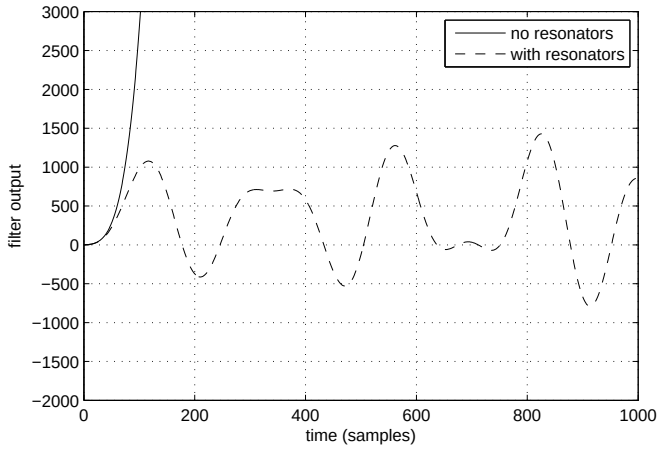


Figure 6.6: First 1000 samples of the impulse response of a 5th order loop filter with a corner frequency of 100 kHz, once with and once without resonator sections (12 kHz and 20 kHz).

In the reasoning above a loop filter without resonator sections was used, of which the impulse response grows constantly over time. In the case of a loop filter with resonators sections the impulse response initially also grows, as shown in fig. 6.4. However, when a larger part of the impulse response is examined (fig. 6.6 and fig. 6.7), it becomes clear that the resonators cause an oscillation with an increasing amplitude in the output. As a result, a single impulse will cause a filter output that alternates between positive and negative values, while the envelope grows over time. It might seem that this will cause problems and that no convergence on the first symbol will be realized, but this is not the case.

Since the impulse response initially grows fast for tens to hundreds of cycles, the decision on a symbol will typically have been made before the impulse response starts to temporarily decrease, and the situation is equal to that of a loop filter without resonators. If the decision on a symbol has not yet been made before the impulse response starts to

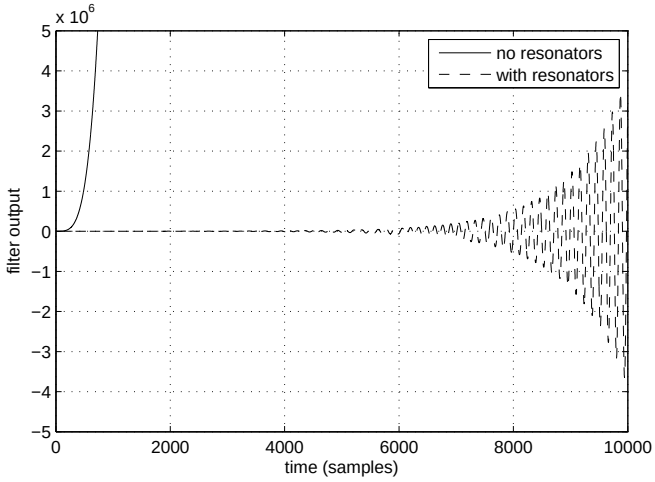


Figure 6.7: First 10 000 samples of the impulse response of a 5th order loop filter with a corner frequency of 100 kHz, once with and once without resonator sections (12 kHz and 20 kHz).

temporarily decrease, the impact of the symbol on the combined output will become smaller. At this moment the impact of more recent trial feedback symbols will be bigger, and the probability that a decision will be forced by those symbols increases. Since the total number of parallel solutions under investigation is limited, this decision is likely to also influence the number of alternatives for older symbols. As a result, eventually a decision will be made for each symbol. This decision will typically be made first for the older symbols and then for the more recent symbols.

Algorithmic structure

The pruned look-ahead SDM algorithm is a specific implementation of the generic look-ahead enabled noise-shaping quantizer, as introduced in sec. 5.1 and schematically depicted in fig. 5.2. The cost function of the system consists of two cost functions in series (see fig. 4.8). The first cost function is the loop filter, which evaluates the frequency distribution of the error signal. The output of the loop filter is passed to the second cost function, i.e. the quantizer cost function (sec. 5.3.1), which calculates the final output value of the cost function. The two cost functions are

applied to all the $2N$ parallel potential solutions, such that $2N$ cost values are resulting. Finally a selection is made to determine which solutions are kept, on the basis of the cost associated with each solution.

Although the high level structure of a pruned look-ahead modulator is comparable to that of a full look-ahead modulator (fig. 5.4), the details differ at a number of points.

In the full look-ahead algorithm only a fraction of all the solutions under investigation accurately describe the input signal. Therefore, there is a main loop filter required that contains the states resulting from the actual selected solution (see sec. 5.3). All the look-ahead filters are at each conversion step loaded with this state, and from this shared starting point alternative solutions are investigated. The solutions that are investigated are generated by the trial feed-back sequence generator, i.e. the sequence generator feeds each look-ahead filter with a different symbol sequence of length N.

In the pruned look-ahead algorithm all solutions under investigation describe the input signal accurately. More specifically, all the solutions converge over time to one and the same path, such that when the output symbol is determined there is only one answer. Because of this property, the state of every look-ahead filter is always resulting from the same output bitstream while the differences between the states of the parallel look-ahead filters are caused by the part of the solution that is not yet determined. Since all the parallel alternative solutions will, by design, convergence on the same final solution, there is no need for a central loop filter that receives the output symbol to update its internal states as is the case in the full look-ahead algorithm.

Because the depth of the look-ahead tree is increased with only one level per clock cycle before it is pruned, the trial feed-back sequence generator that is required by the full look-ahead algorithm is replaced with a trial feed-back symbol generator, which delivers a single symbol per clock cycle to each look-ahead filter. Since the symbol delivered to each look-ahead filter is always the same, i.e. each of the parallel sequences is always extended with a '0' and with a '1', the feed-back symbol generator is simply providing the two symbol values at its output, which are hard-wired to the parallel loop filters.

As a result of the above, the internals of a pruned look-ahead modulator consist only of N parallel look-ahead filters with control, a trial feed-back symbol generator, and an evaluate and select function, as depicted in fig. 6.8. A feed-back path is present from the evaluate and select block to the bank of parallel filters. Note that this feed-back path is

different from the feed-back path of a full look-ahead modulator since it does not indicate what output symbol is selected, but it indicates what N out of the $2N$ generated results are to be kept. The output symbol is determined by finding the trial feed-back symbol that was applied to a filter L clock cycles ago, and cannot be determined on the basis of what paths are selected or what path is the best.

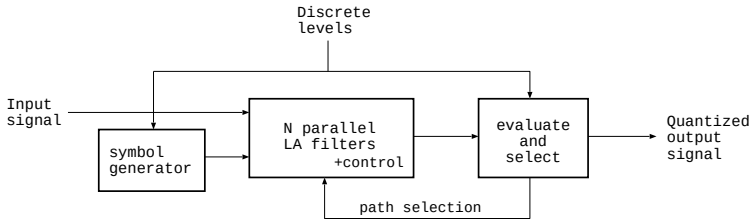


Figure 6.8: Block diagram of a pruned look-ahead modulator, consisting of a symbol generator, N parallel look-ahead filters, and evaluate and select block.

Algorithm steps

The complete generic pruned look-ahead algorithm for 1-bit modulation consists of only four basic steps and is characterized by two independent parameters, i.e. N denotes how many paths are investigated in parallel and L defines the look-ahead depth or latency of the algorithm. The startup and shutdown of the algorithm require special attention and are discussed separately.

In the first step of the algorithm, a selection criterion is applied to the current set of solutions under investigation (N unique solutions with length $L-1$). After this operation the $N/2$ best paths of length $L-1$ that will be expanded are known. The selection criterion (cost function) can take several parameters as input, e.g. the accumulated path cost or (part of) the symbol sequence under investigation, and dictates the type of optimization that is performed (sec. 5.3.1).

In the second step of the algorithm, each of the selected paths will first be duplicated and expanded. The duplication consists of the creation of a copy of the $L-1$ symbols, the filter states, and the accumulated path score of the solution. Next, the paths will be expanded, one of the paths of each pair with a '0' bit, and the other path with a '1' bit. The cost for appending the symbol to the solution is added to the

already accumulated cost for the path. From this point onwards the two solutions are unique and differ in their symbol history, the filter states and the path score. In the next conversion cycle this accumulated cost will be input to the selection criterion. At the end of this step there are again N paths with length L.

The third step of the algorithm consists of the selection of the output symbol. If it is assumed that the latency L is large enough, all paths will have an identical first symbol, and the output symbol for the input sample that was applied L conversion cycles ago has been found. If convergence cannot be guaranteed the output symbol should be selected from the best path and special action is required, as separately discussed in another section.

In the fourth and last step of the algorithm the first symbol is removed from the N solutions, generating the N unique solutions with length $L-1$ that are the input to the first step. No update of a main modulator is required, since each of the parallel solutions has a dedicated look-ahead modulator assigned to it which is updated in the second step. Finally the conversion time can be advanced and the next input sample can be processed.

Start and end of a conversion

Because of the inherent latency of the pruned look-ahead algorithm, both starting and ending a conversion require special attention. At the start of a conversion, no look-ahead tree exists yet. Only after L conversion cycles the full look-ahead tree depth has been realized and the first useable output symbol becomes available. The first $L-1$ output symbols are not resulting from an actual input signal and should be discarded.

A second point that requires attention during startup of the modulator is the initialization of the system. When the initial look-ahead tree is constructed all potential solutions should start from the same situation, i.e. the initial conditions of the N modulators should all be identical. However, when the initial conditions are all identical the modulators will all give the same response, since they are supplied with the same input signal and the same trial feed-back symbols. As a result, if no special care is taken in the selection procedure of the algorithm, it is possible that the same solution will be investigated multiple times by different modulator instances, reducing the effectiveness of the approach. During the first $log_2(N)$ conversion cycles it is therefore mandatory to steer the selection process in such a way that different trial solutions

are selected, even if these are not resulting in the lowest cost. Once the system is completely running and all parallel modulators operate on a different solution, it is not possible anymore that the system will investigate the same solution more than once, and the system will start to select the most promising solutions automatically.

At the end of a conversion, after the last input sample has been applied to the converter, another $L-1$ conversion cycles are required in order to get the last output symbol out of the converter. During those cycles a suitable input signal, e.g. silence, should be applied to the converter such that convergence on the output will be reached. No further changes or precautions are required, and after the $L-1$ additional conversion cycles the complete output sequence has been obtained.

Potential convergence issues

The algorithm for efficient pruned look-ahead modulation requires a large look-ahead depth such that the first symbol of all the solutions will be identical. If the look-ahead depth is not large enough, the algorithm will not realize convergence on the first symbol, and paths that do not match the selected output symbol will have to be removed from the solution search space. In principle the original problem of a varying amount of investigated solutions has returned, although in this case the situation is less severe. Because of the gradual elimination of less favorable solutions, it is reasonable to assume that most of the solutions will start with the same symbol as the best path. As long as at least half of all the solutions agree with the selected output symbol, it is possible to continue with the nominal number of solutions. If less than half of the solutions agree with the proposed output symbol, the validity of this choice is questionable, and it should be considered to change the output symbol, after which it is again possible to extend the solution space to the nominal amount of paths. If it is decided to not change the output symbol choice, temporarily the number of solutions under investigation will be smaller than nominal. It is to be expected that this situation will not occur often, and as a result the number of solutions will be restored to nominal in at most $log_2(N)$ conversion cycles.

The average number of conversion cycles required for reaching convergence is expected to be a function of the amplitude of the input signal. In general there are fewer valid bit sequences for accurately representing input signals with a large amplitude than for small input signals. It is therefore expected that convergence will be reached faster for large signals, simply because there are few reasonable candidate solutions. In

the case of low level amplitude signals many possibilities exist for representing those, and the difference in quality will only become visible after many conversion cycles. It is therefore imaginable that for low amplitude signals no convergence is reached within the look-ahead depth. As long as this situation is detected, e.g. by comparing if the output symbols found by all parallel solutions are equal, and the solutions that do not match the selected output symbol are discarded, it is expected that there will be virtually no degradation of the signal quality compared to what could be realized, since the difference in quality between the solutions was very small. Once the difference in quality between solutions starts to become significant, the inferior solutions will be automatically removed from the look-ahead tree.

The frequency of the input signal is not expected to have any significant influence on the number of conversion cycles required to reach convergence, as long as the oversampling rate of the modulator is high enough. In the case of a low oversampling rate, e.g. less than 32x, higher frequency signals might be more difficult to encode accurately, and this could possibly increase the rate of convergence.

Independent of the input signal, the number of parallel paths influences the time required to reach convergence. When more paths are available, more solutions can be investigated, and the probability of finding paths that match well with the input signal increases. As a result, the probability increases that the difference in quality between paths becomes smaller, and therefore more time will be required to determine which path is the best. Thus, it is expected that a larger latency L is required when the number of paths N increases.

Conversion quality as a function of the number of paths

The number of parallel paths N that is used in the pruned look-ahead algorithm dictates the required computational power and the amount of resources. If more solutions are investigated in parallel, a larger part of the solution space can be explored and more promising solutions can be analyzed over a larger time span, increasing the probability of finding the optimal solution. If less parallel paths are used by the modulator, it is expected that the quality of the solution will be of a lower quality since the probability of finding the best solution is smaller. This will most likely manifest itself as a reduced increase in stability of the converter, more harmonic distortion, and larger errors in the transient response. Still, an improvement over a normal, i.e. not look-ahead enabled, Sigma-Delta converter should be possible, even for a low number of parallel

solutions.

It is expected that the relation between the number of paths and the increase in quality is non-linear. More specifically, when the number of paths is increased, initially a strong improvement in the quality of the output is expected. This improvement will be realized because the decision on the output symbol can now be based on the long term effect of the symbol on the output signal. Once the number of parallel solutions becomes larger, the gain in improvement will become less and will start to flatten out. The number of parallel paths is now large enough to cover nearly all the realistic output sequences. Only in a limited number of cases a decision will be made that is relatively far away from the optimal point. Finally, no improvement is possible anymore and a further increase of the number of paths will not change the result.

The number of paths that is required to reach a (near) optimal result will depend on the input signal characteristics and the cost function. For example, the number of possibilities for representing a large signal is very limited, and with a relatively small number of paths it should be possible to find the optimal sequence or a good approximation to it. In the case of a low amplitude signal, there are many binary sequences that are possible candidates, and only with a very large number of paths it can be expected to find the optimal sequence. However, in this case it is not a major problem when a solution is found that is relatively far from the optimal one, i.e. the selected solution will not cause the converter to go unstable but will probably only cause a small increase in the noise floor.

Since the cost function determines what paths are kept and what are discarded, the cost function also indirectly determines what bit combinations are tested. If the cost function is awkwardly chosen it might discard promising paths and keep paths with a low potential, and the final solution found will be of a lower quality. This problem will show up more severely when a low number of paths is available to the modulator, and only by increasing the number of paths the conversion quality might be improved. Thus, the cost function selection criterion will have a large influence on the number of required parallel paths.

Required resources per parallel solution

In the pruned look-ahead approach there are N solutions which are investigated in parallel. These N solutions are different as a whole, but are identical when only the oldest symbols are considered. The length of the identical part is varying over time and is of no further

interest for the correct functioning of the algorithm as long convergence is reached, i.e. the oldest part of the complete set of solutions should always be identical. Because of these properties it is necessary to store the N sequences with length L all independently. Since each path under investigation is different from all the other paths, also the filter states that correspond to this solution are unique and are coupled to the path. Finally, there is the accumulated path cost which is specific to a solution.

6.2.4 Summary

The introduction of pruning enables larger look-ahead depths without the associated increase in the number of filter output evaluations, as would be needed in the full look-ahead case. However, straightforward application of the pruning concept is only possible when no results are re-used from the previous conversion step. For a higher computational efficiency re-use of results is desired, but this can not be efficiently realized for moderate look-ahead depths. By increasing the look-ahead depth L to a very large value, i.e. hundreds of symbols, without increasing the number of parallel paths N, the problem of a varying number of solutions under investigation has been solved and efficient pruned look-ahead is possible. The cost of the approach is limited to an increase of the converter's latency and the required memory storage, i.e. the latency is equal to the look-ahead depth and the amount of memory required to store the solutions under investigation is proportional to the pruned look-ahead depth and the number of parallel paths. If the look-ahead depth is large enough, it is not required to determine what solution is the best in order to obtain the output symbol, since all solutions will return the same output symbol. The quality of the resulting output is influenced by both the cost function, i.e. the selection criterion that determines what paths are investigated further and what are rejected, and the number of solutions under investigation. It is expected that with a proper cost function only few parallel paths are required in order to improve conversion quality significantly.

6.3 Pruned look-ahead modulator realizations

Based on the insights obtained in the previous chapters, and especially the insights from the previous section, several pruned look-ahead modulator implementations have been realized. These realizations, including a realization of the original Trellis modulator by Kato, are all detailed

in the next chapters. In this section the reasoning that led to these realizations is discussed.

6.3.1 Trellis sigma-delta modulation

The Trellis sigma-delta modulation algorithm [37, 38] is the first published demonstration of a pruned look-ahead approach. Compared to full look-ahead modulation, a significant increase in performance is realized. However, in this algorithm it is still required to investigate a large number of solutions at every time step, making it impractical for actual use. In ch. 7 the algorithm and its performance are investigated in detail.

The algorithm has two parameters. The first parameter, called the Trellis order N in the Kato papers, controls the amount of filter output evaluations per conversion. The second parameter, denoted as t_0 in the original papers but in this work called the Trellis depth L, sets the pruned look-ahead depth and latency. The relation between the parameters is schematically indicated in fig. 6.9. In the original work the definition of N is such that at every time step the number of filter output values to evaluate equals 2^{N+1}. Half of those 2^{N+1} results are discarded and 2^N solutions remain. The selection criterion is such that if only the newest N symbols of each solution are taken into account, all the solutions are unique, i.e. all the 2^N possible sequences of length N are covered. As a result, for each of the 2^N different sequences that are kept a choice between two potential sequences has to be made. From those two potential sequences the solution with the lowest cost is selected. Thus, the pruning selection criterion combines two requirements. Firstly, each symbol sequence with length N should be present exactly once when the newest N bits of all the sequences are considered, and secondly, the selected path should have the lowest accumulated path cost.

The select and discard procedure is repeated until a pruned look-ahead depth of length L has been realized. At this moment, there are 2^N different solutions with a length L, and an output symbol can be determined. In the Trellis algorithm the output symbol is selected by tracing back any of the solutions L time instants, using the Trellis-Viterbi algorithm. From sec. 6.2.3 it is known that such a complicated procedure is not required in order to find the output symbol, but because of how the algorithm was modeled, or possibly because of limited insight in the actual operation of the algorithm, Kato found this approach necessary.

Crucial for the correct functioning of the algorithm is the value of the

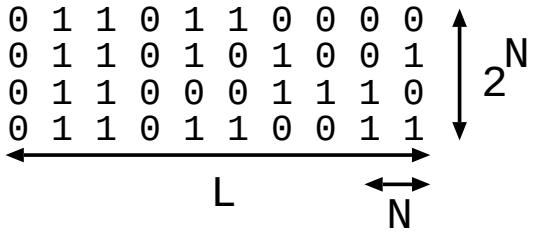


Figure 6.9: Relationship between the Trellis depth L and the Trellis order N. The newest N bits of each of the 2^N sequences are different. The converter output decision is made by tracing back L clock cycles.

selected Trellis depth L. If the length is chosen too short, the system will not be able to converge, and the first symbol of the 2^N parallel solutions will not be uniquely determined. As a result the quality of the conversion will reduce and truncation noise will be present (see sec. 7.4). At the cost of a larger latency and more memory the value of L can be increased, such that the probability of reaching convergence will increase. This increase has no impact on the number of required filter output evaluations, and does not change the outcome of the conversion once the required minimum length has been realized. The conversion performance is thus only a function of the Trellis order N once the minimum required length L has been reached.

6.3.2 Efficient Trellis sigma-delta modulation

In the Trellis algorithm every possible combination of symbols for the last N symbols is evaluated, i.e. 2^N parallel possibilities which can all be realized twice in the case of a 1-bit SDM. It can be easily seen that this scheme, although pruning the solution space tremendously, still forces a large number of unnecessary calculations. For example, if the input signal is small, the average value of the feed-back value should also be small. For such a signal it is very unlikely that the optimal feedback signal consists of many identical symbols, since this would describe a large signal. However, because of the nature of the Trellis algorithm such solutions are evaluated and require resources while they will not change the outcome of the process.

Furthermore, it is highly unlikely that a feed-back sequence that has a bad match with the input signal, measured over a reasonable number

of symbols, will suddenly show a good match when a single symbol is added to it. However, a sequence that shows a good match will typically continue to show a good match when a symbol is appended (sec. 8.1).

Based on these observations it can be expected that a good performance can still be achieved when the Trellis algorithm is changed such that less than 2^{N+1} solutions are investigated per time step. Only the solutions with a high potential, i.e. those solutions that could possibly determine the output symbol, should be investigated. By comparing the cost value of the different solutions it can be determined what solutions are promising and what are not, i.e. the lower the score the higher the potential of the solution. Depending on the input signal there will be less or more solutions that are promising.

In order to select the best solutions it would be possible to set a limit for the cost value which determines if a solution is good or bad. The disadvantage of this approach is that it will result in a time-varying computational load. Furthermore, it will only be possible to obtain good results if a proper threshold value is selected. If the threshold is set too high, too many solutions need to be investigated. If the threshold is set too low, only a small number of solutions will be investigated and non-trivial solutions might be missed. It is therefore more convenient to fix the amount of filter output evaluations, and to always select the best half of the results to continue with.

Compared to the original Trellis algorithm, the efficient Trellis algorithm [20] requires a third parameter M, which determines the number of solutions under investigation. The Trellis order N is still required, but now only determines how many of the last bits of the M parallel solutions are required to be different. The Trellis depth L again determines the maximum pruned look-ahead depth and the latency of the modulator.

The selection criterion that is applied to the set of solutions is different from that of the original full Trellis algorithm. At each conversion cycle the best M of the $2M$ solutions are selected to continue, where best is interpreted as having the lowest cost. Identical to the operation of the Trellis algorithm, the solutions that continue should all be different in their newest N symbols. If two solutions have their newest N bits identical, only the solution with the lowest cost is kept and the other is removed from the set of solutions. Since a total of M valid solutions is required, the best solution of the ones that were not selected will be added to the set to get back to M solutions.

The output symbol is determined with a latency of L conversion cycles. If any of the M paths has not converged on the selected output symbol,

this path will receive a large cost penalty, such that the path will be automatically removed from the solution space because of the select and discard operation. This additional check for convergence solves the problem of truncation noise and enables the use of a shorter Trellis depth.

Since only a fraction of the original 2^{N+1} solutions are investigated, the likelihood of finding the optimal result decreases. However, because less unnecessary trial sequences are tested it is possible to obtain similar results to the full Trellis algorithm with $M << 2^N$. Furthermore, in general the quality difference between the best and one-but-best solution should be small, and therefore only a small performance penalty is expected. When less and less solutions are investigated the quality of the generated bitstream should degrade back to the quality of a normal SDM.

In ch. 8 the Efficient Trellis algorithm is detailed and the signal conversion performance as well as the computational load are evaluated.

6.3.3 Pruned Tree sigma-delta modulation

In the Efficient Trellis algorithm (ch. 8), a relatively large amount of operations is required to check and guarantee that the newest N symbols of the M solutions are unique. Furthermore, if N is chosen too small, a negative effect on the signal quality performance is resulting in some cases. This reduction in quality is caused by the removal of one of two similar solutions that have their newest N bits identical. The Efficient Trellis algorithm will select and keep the solution with the lowest cost and remove the other. However, if both solutions are very similar over a large number of bits, it is likely that their cost values are also very similar. It is in this situation possible that the solution that is removed would have developed to a better solution than the solution that is maintained. Thus, from a signal conversion perspective there is a clear motivation to have N large, preferably equal to L. This however increases the computational load and reduces the efficiency of the Efficient Trellis approach.

From the discussion above it is clear that there are opposing constraints on N, i.e. on the one hand N should be large, but in order to minimize the computational load N should be small. Fortunately, there is a solution to this problem which combines the benefits of a large and a small N, namely the removal of the check for unique sequences. The resulting algorithm is called pruned tree sigma-delta modulation.

Removing the check for maintaining unique solutions is not without danger. It should be guaranteed that under all circumstances the M solutions under investigation will not be identical. Once solutions become identical, the number of solutions that is being investigated effectively reduces by one, and useless operations are performed. At first sight it seems therefore that a check for maintaining unique solutions is required. However, it can be shown that with a proper initialization of the system no additional checks are required and that M unique solutions will be maintained. In fact, it is even possible that temporarily two solutions have their newest L symbols identical, but since both solutions have reached this situation via a different route the cost for the two solutions will be different, and the effect of adding the same symbol to both solutions will be different. As a result, the solutions will start to differ and eventually one of the two solutions will be rejected because it is too expensive.

Because the check on uniqueness of the symbol sequences is removed, the pruned tree algorithm requires only two parameters. In the original publication on this algorithm [32] the parameter N denotes the number of parallel paths that are kept active in the system, but in this thesis M is used, equal to M of the Efficient Trellis algorithm. The parameter L sets the latency of the algorithm. Note that in the publication the algorithm is called Efficient Trellis algorithm because of marketing reasons, while in reality there is little resemblance with the (efficient) Trellis algorithm.

The pruned tree algorithm is a practical realization of the pruned look-ahead algorithm with reuse, as described in sec. 6.2.3. In each conversion cycle all the M parallel solutions are first extended with a '0' and a '1' bit. Then, from those resulting $2M$ solutions the M best solutions are selected and passed to the next conversion cycle. The output symbol is determined with a latency of L cycles from the best path, and a test is performed to make certain that the other solutions have converged. Although this last operation is not required in theory, it is included since the computational overhead is small and a more robust system results. Furthermore, for practical reasons an adjustment of the path cost is performed to prevent values from increasing to infinity. The selection criterion for determining what paths continue is only acting on the accumulated path cost. Good performance has been realized with the cost function equal to the filter output squared, i.e. the energy of the frequency weighed error.

The details of the pruned tree algorithm and its performance are discussed in depth in ch. 9.

6.3.4 Pruned Tree sigma-delta modulation for SA-CD

The pruned tree Sigma-Delta algorithm described in ch. 9 shows great improvements over the Trellis based algorithms. The amount of computations per sample is lower, and less parallel paths are required to obtain the same level of signal conversion performance. With a limited number of parallel paths near perfect signal conversion performance is achieved. The only remaining signal conversion quality issue is a limited amount of in-band noise modulation.

A modulator that is targeted for use in SA-CD applications does not only need to provide high quality audio conversion, but should also generate bitstreams that are compatible with the SA-CD lossless data compression algorithm [28, 34, 39, 47]. Application of the lossless data compression algorithm is required in order to fit 74 minutes of audio in both stereo and multi-channel format on a disc. Since the compression is lossless, the compression gain is a function of the input data, which in this case is the 1-bit encoded audio. Thus, a modulator that is ultimately suited for SA-CD applications should not only convert the audio input signal with a high quality, but it should at the same time generate a bitstream that can be compressed well.

Data that can be compressed well contain a high degree of predictability, i.e. repetitive structures. An SDM that realizes a clean output spectrum, i.e. a spectrum that contains no tones, typically generates a bitstream with little correlation. When data contains little correlation it is not predictable, and therefore the compression gain of such data will be low. As a result, a modulator that realizes a high audio conversion quality will typically generate bitstreams that result in a low compression gain.

In the case of a normal SDM it is not possible to optimize the output bitstream for both signal conversion quality and compression gain at the same time, especially since the two criteria are often opposing. In the case of a look-ahead modulator, because of the increased stability, it is possible to optimize both the quality of the output signal and the potential for compression by combining both criteria into a cost function. This cost function should be a weighed combination of two cost functions, one for measuring the signal conversion quality, and one for measuring the correlation of the signal. The cost function that measures the correlation of the output signal should not depend on the input signal, but should be based only on the bit sequence that is being evaluated. Since the SA-CD format was designed to provide an extremely high audio quality, the weighing of the two values should be such that the conversion quality is the main criterion for the optimization, and only when the conversion

quality is not reduced the impact on compression should be taken into account. Thus, the audio encoding quality should be considered as a hard constraint, while the compression gain should be considered as a soft constraint. Such a combined optimization is possible, and some of the results obtained with such a modulator are published in [28, 34]. A detailed description of the approach is given in ch. 10.

6.4 Conclusions

The amount of computations required for performing full look-ahead modulation, compared to the straight-forward brute-force approach, can be significantly reduced by changing the algorithm for calculating the path scores (sec. 6.1). For large look-ahead depths (20 or more) the computationally most efficient solution is to select half of the solutions from the previous clock cycle and to continue from there. This approach reduces the number of filter outputs to calculate at every time step to 2^N, a factor N less than required by the brute-force approach. However, because the number of computations still doubles with every increment of the look-ahead depth, it is practically not feasible to go to much larger look-ahead depths than 20 using this approach.

An alternative solution to reduce the amount of computations per output sample is the introduction of pruning (sec. 6.2). Instead of performing an exhaustive search of the solution space, heuristics are applied to detect promising solutions. This approach enables the investigation of a larger look-ahead depth at a reduced computational cost, resulting in an improved conversion result. The disadvantage of the basic pruning concept is that it is not possible to efficiently reuse results from the previous conversion step, which is a necessity for realizing a very large look-ahead depth at minimal computational costs.

Highly efficient pruned look-ahead can be realized by increasing the look-ahead depth L to a very large value, i.e. hundreds of symbols, without increasing the number of parallel paths N. By applying a select-and-continue strategy which selects the best half of the solutions under investigation, the set of solutions will gradually converge to one solution and the output symbol will be determined. Because all solutions find the same output symbol, a constant number of solutions can be kept active, and reuse of results is possible. With a proper cost function it is expected that with a limited number of parallel paths significant improvements in signal conversion quality can be realized, at a fraction of the cost of full look-ahead modulation.

In order to verify the ideas described in this chapter, several implementations of pruned look-ahead modulators have been realized. The reasoning behind these different realizations is given in sec. 6.3. The details and performance of the different realizations are described in the next chapters.

Chapter 7

Trellis sigma-delta modulation

Trellis Sigma Delta Modulation is the first published look-ahead sigma-delta modulation technique that is able to look-ahead more than a few symbols at reasonable processing cost. The algorithm is first described in a Japanese report [37]. In [38] a summary of the report is presented, which has triggered several authors to investigate new ways of generating high quality bitstreams.

In the Trellis sigma-delta modulation algorithm the traditional SDM feedback structure that attempts to minimize the instantaneous frequency weighted error signal, i.e. the quantizer input, is replaced with a structure that attempts to minimize a specific cost function. The cost function is, typically, the global frequency weighted error signal. This minimization of the global frequency weighted error signal is accomplished by investigating a set of 2^N pruned solutions with a length of L bits in parallel, and by using a search algorithm to determine the best solution.

In the solution from Kato, described in sec. 7.1, the search algorithm is based on the Viterbi algorithm, which is traditionally used for decoding convolutional codes [16,62]. However, the Trellis sigma-delta modulation algorithm can also be mapped to the pruned look-ahead framework that was introduced in ch. 6, resulting in a more efficient realization. A description of the algorithm on the basis of this approach is given in sec. 7.2. Because a Trellis SDM is a specific realization of a generic look-ahead SDM, the theory describing a look-ahead SDM is also valid for a Trellis SDM, and the linearized NTF and STF from sec. 5.6 are verified against measurements in sec. 7.3. From ch. 6 it is known that a pruned look-ahead modulator requires a large look-ahead depth to

be able to unambiguously determine the output symbol at every clock cycle, and that there are several factors that influence this minimally required depth. These relations are very difficult to mathematically derive since they are based on properties of the signal that is converted. Therefore, by means of simulations it is determined what Trellis depth is required (sec. 7.4). Next, in sec. 7.5 the functional performance of a Trellis SDM as a function of its design parameters will be investigated. Before concluding the chapter in sec. 7.7, a number of implementation aspects that are key for realizing an efficient realization are discussed in sec. 7.6.

7.1 Algorithm - Kato model

The Trellis sigma-delta modulation algorithm is a pruned look-ahead algorithm in which 2^N solutions of length L are investigated in parallel. At every time step each of the solutions is extended with a single symbol, resulting in 2^{N+1} new potential solutions. From the 2^{N+1} potential solutions a selection is made, based on the cost of the potential solutions, such that again 2^N solutions remain that are all unique when only the newest N bits are considered. A high level block diagram of Trellis SDM is shown in fig. 7.1.

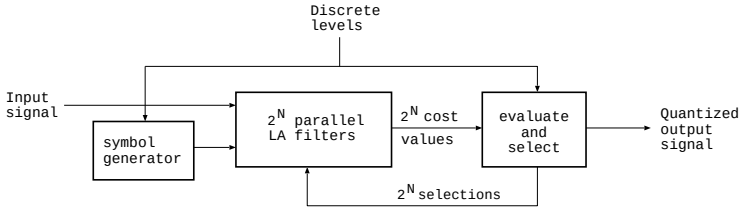


Figure 7.1: Block diagram of a Trellis SDM of order N.

In the Trellis sigma-delta modulation solution, as introduced by Kato, look-ahead is realized by delaying the decision on the output symbol L conversion cycles. The delay is introduced by tracing back L samples in time using the Viterbi algorithm. Although Kato talks about delaying the decision on the output, the effective result of this procedure is the realization of a pruned look-ahead depth of L samples.

The use of the Viterbi decision algorithm implicitly implies that the system which generates the potential output symbols should be modeled with a hidden Markov model (HMM), i.e. the system is assumed to be

a Markov process with an unobservable state. Since the HMM is key for the operation of the Trellis sigma-delta modulation algorithm, this is first discussed before the actual algorithm steps are presented.

7.1.1 Hidden Markov model

In the Trellis sigma-delta modulation algorithm the set of all possible unique output sequences (candidates) are modeled as the states of a hidden Markov model. An output sequence is defined as the last N bits of the running converter output, where N is the Trellis order. In other words, at every time instant there are 2^N possible output sequences that evolve from the 2^N possible output sequences from the previous time instant.

In fig 7.2 the relationship between L and N is schematically illustrated for Trellis order $N = 2$ and Trellis depth $L = 10$. In this example there are $2^N = 4$ bitstreams of length $L = 10$ bits that have (at least) the newest $N = 2$ bits different. The remaining $L - N = 10 - 2 = 8$ bits can be the same but do not need to, except for the oldest (left most) bits that will converge to the same value over time. The bit sequence before the newest N bits reflects the most recent $L - N$ decisions taken to reach the Markov state.

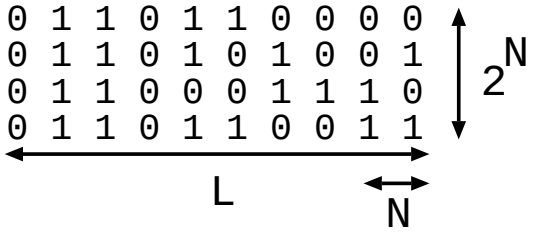


Figure 7.2: Relationship between the Trellis depth L and the Trellis order N. The newest N bits of the 2^N sequences are different. The converter output decision is made by tracing back L clock cycles.

The hidden Markov model can be applied as follows. Consider the set of all binary sequences with length N, consisting of 2^N unique sequences. This situation reflects the starting condition at time $n = t_0$. Now the sequences in this set are extended to length $N + 1$. This reflects the situation at time $n = t_0 + 1$, where each possible sequence is appended once by a '0' and once by a '1' as shown in the left half of fig. 7.3. If

from this new set with length $N+1$ only the newest N symbols are considered, every possible length N sequence is present twice. These newly generated sequences with length N are the new output candidates. In other words, if the 2^N sequences are the 2^N Markov states, there are two possibilities to enter every state. A schematic representation of this dependency results in a Trellis diagram, as shown in the right half of fig. 7.3 for Trellis order $N = 2$. From state '00' it is possible to reach state '00' (via '000') and state '01' (via '001'). Alternatively, it is possible to reach state '00' from state '10', and to reach state '01' from state '10'.

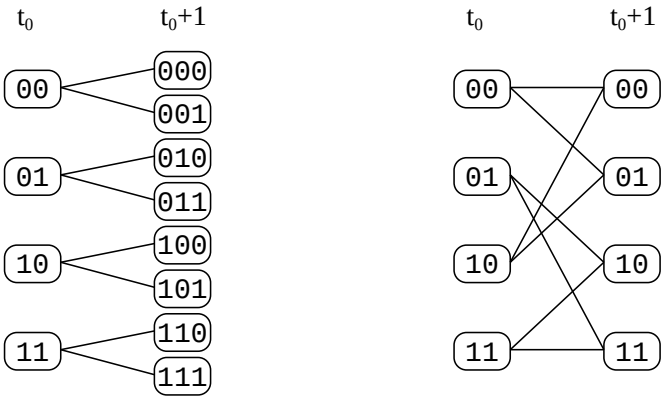


Figure 7.3: Left: Extended states for $N = 2$. Right: Trellis diagram for $N = 4$.

Since for every state there are two possible originating states, a decision on which actual state to use is required. This state transition decision is based on the cost associated with the transition, i.e. the transition with the smallest path metric is accepted. From sec. 5.3.1 it is known that numerous cost functions are possible. In [38] Kato defines the cost function as the square of the frequency weighted error signal, i.e. the filter output squared. By integrating this cost from the start of the conversion (time $n = 0$) to the current time step (time $n = t_0$) the path metric is obtained:

$$\begin{aligned} C(t_0) &= \sum_{n=0}^{t_0} c(n) \\ &= \sum_{n=0}^{t_0} w^2(n) \end{aligned} \tag{7.1}$$

Since the cost function is based on the filter output it is not only depending on the current filter input, but also on the history of the filter. Because of this history, every Markov state contains in addition to the path metric, i.e. the total cost to reach the state, and a link to the originating state, the state variables of the Trellis SDM filter since these encode how the state was reached. The link to the originating state, i.e. the state in the previous clock cycle from which the current state was reached, is required in order to be able to trace back in time to find the originating state L clock cycles ago.

7.1.2 Algorithm steps

The Trellis sigma-delta modulation algorithm consists of five main steps. By executing all steps once, time will be advanced from $n = t_0$ to $n = t_0 + 1$, i.e. one clock cycle, and the modulator's output symbol for the input sample of L clock cycles ago will be found.

Step 1: calculate the cost for appending a bit

For every state $\{s \in \mathbb{N}_0 : s < 2^N\}$ the cost for appending a '0' and a '1' to the generated bitstream is calculated, resulting in a total of 2^{N+1} cost values. These cost values can be divided in 2^N values $c_0(s, t_0)$ resulting from the '0' symbols and 2^N values $c_1(s, t_0)$ resulting from the '1' symbols.

In order to perform these calculations, for every HMM state a look-ahead filter structures is required, i.e. 2^N parallel structures in total. Each structure calculates on the basis of its internal state, the input signal $x(k)$ and the feedback value $fb(k)$ a cost value $c(k)$. In the example look-ahead filter realization of fig. 7.4 that is based on a a feed-forward loop filter, the cost signal $c(k)$ can be recognized as the filter output $w(k)$ squared.

Step 2: calculate the path metric

For every state s, the accumulated cost value (path metric) for the complete trial output sequence is calculated. Recall that the path metric is the sum of all the cost values required to reach the current time step (eq 7.1). By induction it follows that the cost values for reaching the two next possible states from state s are given by

$$C(s, t_0)_0 = C(s, t_0) + c_0(s, t_0) \tag{7.2}$$

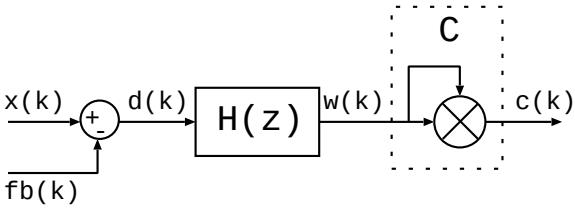


Figure 7.4: Look-ahead filter structure with feed-forward loop filter and concatenated cost function.

$$C(s,t_0)_1 = C(s,t_0) + c_1(s,t_0) \tag{7.3}$$

where $C(s,t_0)_\sigma$ represents the total cost required to reach state s at time $n = t_0$ and append a trial value σ to the bitstream, $C(s,t_0)$ is the cost for reaching state s at time $n = t_0$, and $\sigma \in \{0,1\}$. Thus, the total cost (path metric) is the sum of the running path cost increased with the cost involved for continuing the path with the trial feedback symbol. The output of this operation consists of 2^{N+1} path metric values.

Step 3: sequence selection

From the 2^{N+1} possible output sequences a selection is made which of those are progressing to the next time step. For every state s at time $n = t_0 + 1$ there are two originating candidate states at time t_0. Fig. 7.5 illustrates this dependency for the case $N = 2$, and for the general case, where ω_N represents a bitstream of length N.

To ease notation, without loss of generality, the binary representation of state number s will be taken equal to ω_N. The two states (paths) that connect to state $\omega_{N-1}\sigma$ are states $0\omega_{N-1}$ and $1\omega_{N-1}$. The state with the smallest path metric is selected as source state and will move to state $\omega_{N-1}\sigma$ in the next time step. The state variables of the target state $\omega_{N-1}\sigma$ will be replaced with the updated information from the source state. The information contained in the state includes the new path metric, the previous state number, and the updated filter state.

Denote the two source states by

$$\begin{aligned} source0 &= 0\omega_{N-1} \\ source1 &= 1\omega_{N-1} \end{aligned} \tag{7.4}$$

The cost for reaching the target state from the two sources states $source0$

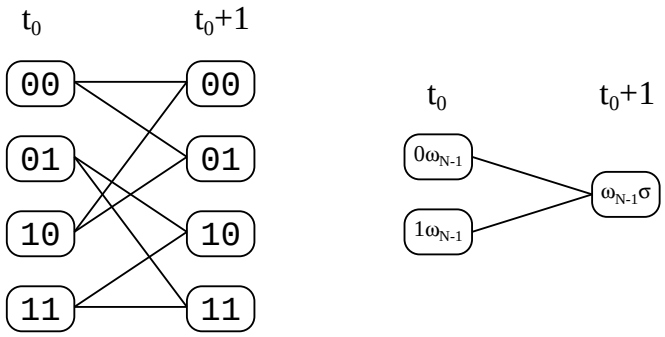


Figure 7.5: Left: state dependency diagram for $N = 2$. Right: generic state dependency diagram.

and $source1$ is given by

$$\begin{aligned} C_{source0} &= C(source0, t_0)_\sigma \\ C_{source1} &= C(source1, t_0)_\sigma \end{aligned} \tag{7.5}$$

The new path metric for target state $\omega_{N-1}\sigma$ is given by

$$C(\omega_{N-1}\sigma, t_0 + 1) = \begin{cases} C_{source0} & \text{if } C_{source0} \leq C_{source1} \\ C_{source1} & \text{if } C_{source1} < C_{source0} \end{cases} \tag{7.6}$$

The new loop-filter state variables for state $\omega_{N-1}\sigma$ for time $n = t_0 + 1$ are obtained by applying σ to the feedback input of the filter from the previous state ($n = t_0$) and updating the state variables.

Step 4: bounding the path metric

Since the path metric is calculated by integrating all the cost values, starting at time $n = 0$, it is a monotonically increasing value. Therefore, for a practical realization, it is required to adjust the path metric values on a regular basis to keep them from overflowing. Since the path selection process is not based on the absolute value of the path metric, but only on the difference between two values, the path metric values can be adjusted by an arbitrary amount without influencing the decision outcome. By subtracting a properly chosen value from all the path scores, the values can be made bounded and the problem is solved. A practical

solution is to subtract the smallest path metric from all the score values, such that all the values are positive and minimal. The maximum value that the path scores can obtain depends on the number of parallel solutions N, i.e. more parallel paths will result in a larger difference in scores and thus a larger maximum value.

Step 5: output code selection

The last step of the algorithm is the generation of the converter output code by means of the Viterbi algorithm. In contrast to a normal SDM, a Trellis SDM will output at time $n = t_0$ the output symbol for time $n = t_0 - L$, where L is a constant pre-defined latency known as the Trellis depth L. A large enough latency is required for unambiguously determining the output code.

The procedure to recover the output is as follows. Starting at time $n = t_0$ at *any* of the states, it is determined from which state at time $n = t_0 - 1$ the state originated. For this state at time $n = t_0 - 1$ it is again determined from which state at time $n = t_0 - 2$ it originated. This procedure is repeated until time $n = t_0 - L$ is reached. The symbol σ that was required to go from $n = t_0 - L - 1$ to $n = t_0 - L$ is the recovered output code. Independently of what state was used to start the recursive backward search the same output code will be recovered, under the condition that L is large enough.

As an example, the process of tracing back from time $n = t_0$ is illustrated in fig. 7.6 for Trellis order $N = 2$. The thick lines indicate the traced back state transitions. Independently of the starting state at $n = t_0$, tracing back to $n = t_0 - 3$ results in arriving at the state '10'. The path, and therefore the output codes, for $n < t_0 - 3$ are uniquely defined. The recovered output code for $n = t_0 - 4$ equals '0'.

Fig. 7.7 illustrates the result after advancing the time by two periods. The thick lines again show the recovered path. The striped lines show the previously recovered paths that have now been rejected by the algorithm. By advancing time two steps, the path up to $n = t_0$ has been uniquely determined. It is clear that that the convergence point can change as a function of the input signal.

In practice, the required trace back depth for unambiguously determining the output symbol can be very large, up to 100s or even 1000s of bits, depending on the Trellis order, the input signal and the loop-filter architecture. An insufficient Trellis depth will result in reduced performance of the algorithm since the output symbol cannot be uniquely

determined. Without special precaution this situation will result in an increased noise-floor, caused by so-called truncation-noise. An investigation on the required Trellis depth as a function of various parameters is made in sec. 7.4.

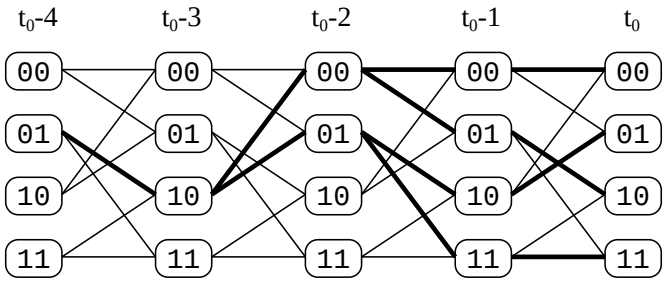


Figure 7.6: Independently of the starting state, tracing back from time $n = t_0$ recovers the state dependency up to $n = t_0 - 3$. The thick lines indicate the recovered state dependency.

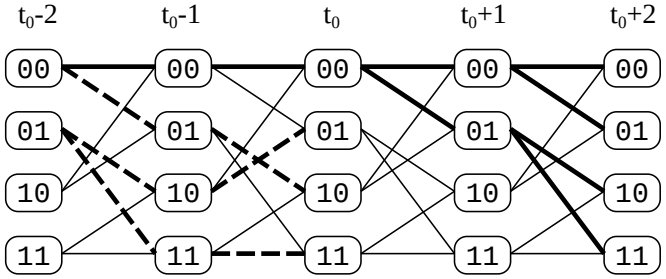


Figure 7.7: Advancing time two steps recovers the state dependency up to $n = t_0$. The thick lines indicate the recovered state dependency. The striped lines indicate the at time $n = t_0$ recovered sequence (fig. 7.6).

7.2 Algorithm - pruned look-ahead model

Although the Trellis sigma-delta modulation algorithm as published by Kato is based on trace back using the Viterbi algorithm, the Trellis sigma-delta modulation algorithm can be implemented equally well us-

ing the pruned look-ahead model of sec. 6.2. More specifically, by realizing a pruned look-ahead modulator that is able to reuse results from the previous clock cycle (sec. 6.2.3), and by selecting the correct cost function, a Trellis modulator will result.

In the Trellis algorithm there are two parameters, i.e. Trellis order N and Trellis depth L, that control how pruning of the solution space is performed. The Trellis order N has a two-fold impact on the algorithm. Firstly, the number of parallel solutions that are maintained at each time step is fixed to 2^N. Secondly, when selecting the 2^N solutions that continue from the 2^{N+1} possible, there is the requirement that the selected solutions are all different when compared over the newest N symbols, i.e. all 2^N possible sequences of length N are covered. As a result, the pruning operation does not select the 2^N solutions with the lowest path metric, but the pruning selects 2^N times the best solution from two possible solutions. Thus, the pruning selection criterion combines two requirements. Firstly, each symbol sequence with length N should be present exactly once when only the newest N bits of all the sequences are considered, and secondly, the selected path should have the lowest accumulated path cost. As a result of this selection criterion the sum of all the path metric values of the 2^N remaining paths will, typically, not be minimal.

In the implementation of Kato, the Trellis depth L defines how far back in time the Viterbi search should go. For the pruned look-ahead algorithm this translates to the pruned look-ahead depth of the converter. Thus, once the pruned look-ahead depth of L symbols has been reached the first output symbol becomes available. Under the assumption that L is large enough, the oldest symbol of all the 2^N solutions will be identical and can be selected as the output symbol. If L is not large enough, also with this approach, truncation noise will be added. The advantage compared to the original Trellis algorithm is that it is now possible to detect the absence of convergence and perform a smart selection of the output symbol in order to avoid the truncation noise (see sec. 7.6.3. Furthermore, since no back tracking is required the computational efficiency of the approach is higher.

In summary, the Trellis sigma-delta modulation algorithm can be efficiently realized by mapping it to the generic pruned look-ahead framework. In order to obtain a functionally identical performance 2^N parallel solutions should be maintained over a pruned look-ahead depth of L. When pruning the solution space the selection should be made such that the 2^N solutions that remain are all different in their last N symbols. This can be realized by choosing between the proper two solutions 2^N

times. Because no Viterbi search is required the computational efficiency of the Trellis algorithm if implemented as pruned look-ahead modulator will be higher.

7.3 Verification of the linearized NTF and STF

Since a Trellis SDM is is a specific realization of look-ahead modulator, the linear model of a generic look-ahead SDM (sec. 5.6) is also valid for a Trellis SDM. The STF and the NTF that are predicted by the linear model are verified by comparing them to actual measurements. For these experiments a fifth order SDM with resonators sections is used.

7.3.1 NTF

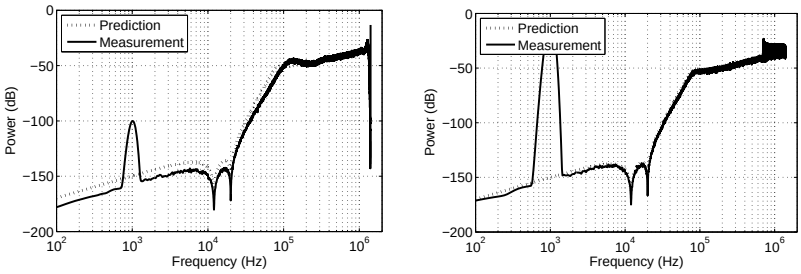
In fig. 7.8, for two different signal levels, the output spectrum of a Trellis SDM with $N = 8$ is compared to the noise spectrum as predicted by the linear model.

In fig. 7.8(a) the comparison is made for an input signal with an amplitude of -100 dB. The shape of the prediction follows the actual shape of the output noise, but there is a mismatch in the levels. In the baseband region the predicted baseband noise is around 8 dB higher than the actual measured noise, whereas in the high frequency region the measured noise levels are higher. Close to Nyquist there is a relatively strong tone in the output, which is not predicted by the model, since the linear model assumes white quantization noise that is shaped. A significant amount of noise power is stored in this tone, and since the total output power is constant the lower baseband noise-floor is resulting.

In fig. 7.8(b) the same comparison is made for an input level of -6 dB. In this case there is a very good match between the prediction and the measurement. In the baseband region the linear model only predicts slightly more noise than actually present. In the mid and high frequency region the curve of the prediction is hardly distinguishable from the measurement result, but a close inspection reveals that the predicted noise is also here slightly higher than actually realized. The surplus of noise is again stored in frequencies close to Nyquist where multiple tones are present which are not predicted.

From the experiments it can be concluded that the general shape of the noise as predicted by the linear model of a Trellis SDM is accurate, except for the high frequency region where tones are present. However, the absolute level that is predicted can be off significantly, depending on

the signal level that is applied. Accurate SNR predictions can not be made on the basis of the linear model, and simulations are required to get reliable values.



(a) -100 dB (b) -6 dB

Figure 7.8: Comparison of the predicted noise spectrum and the actual output spectrum of a Trellis SDM with $N = 8$ for a -100 dB input signal (a) and a -6 dB input signal (b).

Instead of comparing the predicted and realized noise spectrum of a Trellis SDM, it is also interesting to compare the noise spectrum of a normal SDM and a Trellis SDM. The linearized model predicts a quantization noise spectrum which is very similar, but not identical, to that of a normal SDM when the same loop filter is used. The predicted effect of this difference is an increase in the baseband noise for the Trellis SDM, caused by the different noise-shaping characteristic. Actual measurements confirm this behavior, as illustrated in fig. 7.9 where the spectrum of a normal feed-forward SDM and a feed-forward Trellis SDM ($N = 8$) are shown. Clearly the baseband noise of the normal SDM is lower, which results in an approximately 3 dB higher SNR. For frequencies in the transition region the noise level of the SDM is much lower than that of the Trellis SDM, instead of similar as predicted. Thus, in the mid to high frequency region the SDM output spectrum does not follow the predicted curve. Since the total output power is constant, a very strong high frequency tone is present in the SDM output that compensates for the lower noise power.

7.3.2 STF

The linearized STF equations are verified by comparing the predicted STFs with the actual measured STFs. The measurement stimuli are sine waves with frequencies between 1 kHz and 800 kHz and an amplitude

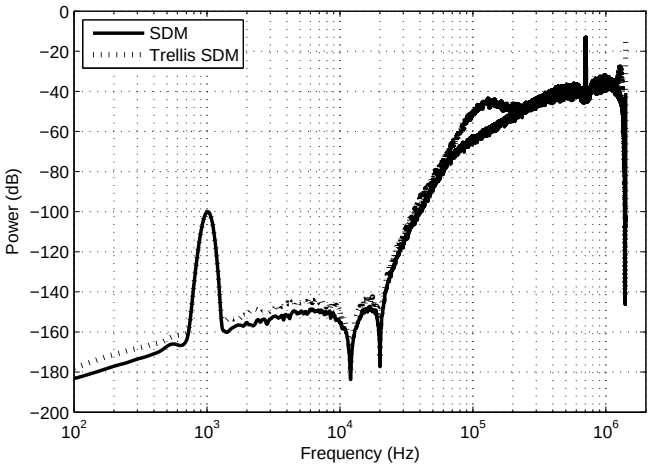


Figure 7.9: Output spectrum of a Trellis converter with $N = 8$ and a normal SDM, both with the same feed-forward loop filter.

of -6 dB. By dividing the measured signal output power by the input power the STF magnitude transfer is constructed. In fig. 7.10 the results are depicted.

For the feed-forward Trellis SDM the predicted STF and measured STF match with great accuracy over the complete frequency range, i.e. the measured STF is very close to unity for frequencies up to 800 kHz. Thus, the mathematically derived linearized STF of a feed-forward Trellis SDM matches with the actual STF.

In the case of the feed-back Trellis SDM the actual realized STF deviates significantly from the predictions for high frequencies. Instead of the predicted peaking an attenuation is realized for frequencies around 40 kHz. The behavior for frequencies above the filter corner frequency also differs from the prediction, i.e. the falloff is less than predicted. For frequencies above 300 kHz it is not possible to distinguish the test tone from the quantization noise, making it difficult to determine the STF. The cause of the mismatches between the predicted STF and the actual realized STF is unclear. As a result, the linearized STF of a feed-back Trellis SDM should not be relied upon and simulations should be used instead to derive the STF.

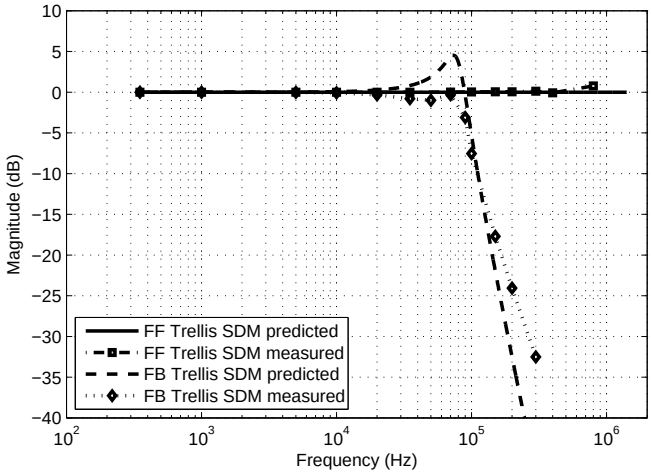


Figure 7.10: Comparison of the predicted and the measured STF of a feed-forward (FF) and feed-back (FB) Trellis SDM.

7.4 Relation Trellis order and Trellis depth

In the paper of Kato [38] the relationship between the Trellis order and Trellis depth is not investigated. However, it is very important to have the Trellis depth large enough, otherwise truncation noise will be resulting. As an example, in fig. 7.11 the output spectrum of a Trellis SDM with $N = 6$ for a 1 kHz sine wave with an amplitude of -60 dB is shown twice. The first curve is obtained for a Trellis depth of $L = 100$ bits, the second curve is obtained for a Trellis depth of $L = 1000$ bits. The curve corresponding to $L = 100$ clearly illustrates how the truncation noise degrades the signal quality.

From the previous sections it is clear that the Trellis algorithm is a heuristic process and that it is difficult to derive the influence of the Trellis order on the required Trellis depth mathematically. Therefore, by means of simulations the relation between the SDM loop filter, the Trellis order, the signal level, the signal frequency, and the required Trellis depth is determined. All the experiments are performed without dither, since from sec. 7.5 it will become clear that there is little need for dithering a Trellis SDM.

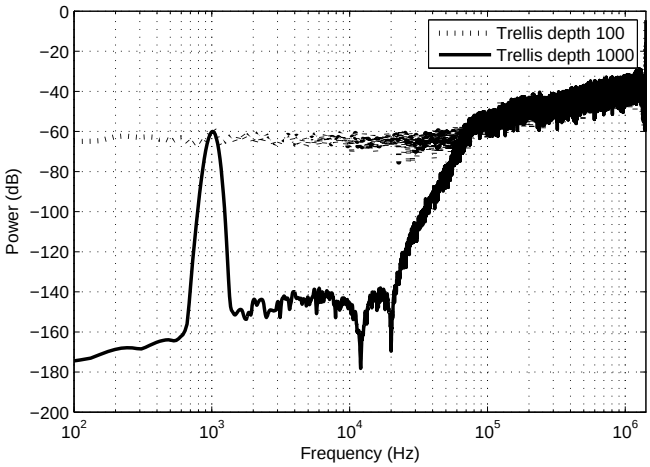


Figure 7.11: Example output spectrum of a Trellis converter with $N = 6$ for Trellis depths $L = 100$ and $L = 1000$. The high noise-floor for $L = 100$ is caused by early truncation of the history.

7.4.1 Simulation setup

In the experiments described next, the actual required Trellis depth for unambiguously determining the output at any time instant, denoted as the observed history length, is determined and stored. This is realized by running the Trellis converter with a very large history length L, such that convergence of all the parallel solutions is always realized. At every moment in time all the parallel solutions are compared in order to determine at which point convergence of all the solutions has occurred. The number of time steps (samples) required to reach this point is the observed history length. Over time the point of convergence varies, as can be seen from fig. 7.12 that shows an example graphical representation of the stored values. As a post-processing step the maximum, the minimum, and the average required Trellis depth are calculated. From the figure it is also clear that the observed history length varies wildly over time, and that reliable statistics can only be obtained if enough samples are processed. As a compromise between simulation time and accuracy a simulation length of $1 \cdot 10^6$ samples is selected.

Several different SDM configurations with varying noise-shaping characteristics are used in the experiments. The details of these configurations

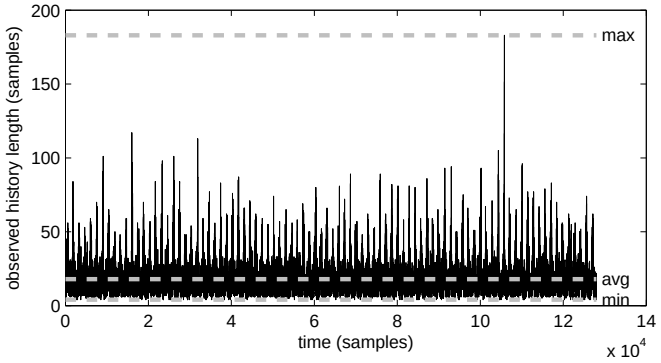


Figure 7.12: The required Trellis depth varies over time. The maximum, minimum, and the average values are indicated.

are listed in app. B.

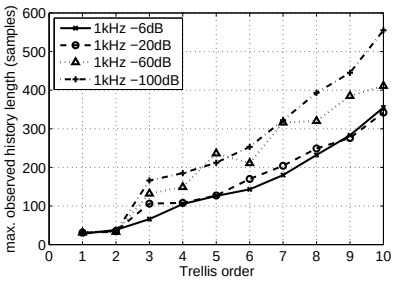
7.4.2 Trellis depth as a function of the Trellis order and the signal amplitude

In ch. 6 it was concluded that the required history length increases with the look-ahead depth N. In the same chapter it was also concluded that the signal level influences the latency required to obtain convergence, i.e. for low amplitude signals a larger Trellis depth L is required than for high amplitude signals. In order to verify these hypotheses the required Trellis depth for encoding a 1 kHz sine wave is measured for Trellis depths $N = 1$ up to $N = 10$. The experiment is repeated for several amplitude levels (-6 dB, -20 dB, -60 dB, and -100 dB). The SDM configuration (SDM1, a fifth order loop-filter with a corner frequency of 100 kHz, see app. B) is kept constant during this experiment.

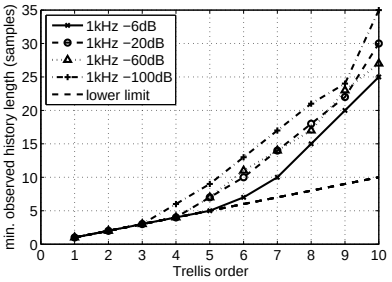
In fig. 7.13(a) the maximum observed history length is depicted. As predicted, for the same signal the maximum observed history length increases with the Trellis order N. For N larger than 2 the signal amplitude has an influence on the required history length, i.e. a larger maximum history length is found for smaller amplitudes.

Instead of only looking at the maximum observed history length, it is also illustrative to look at the minimum observed history length (fig. 7.13(b)). The minimum observed unique history length increases with increasing Trellis order and is larger than the theoretical minimum value for $N > 5$,

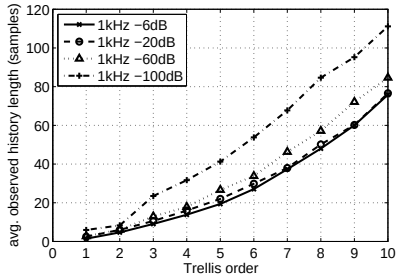
indicating that it is not trivial to select the best solution. There is a small influence of the signal amplitude on the minimum required history length, although the variation is less than for the maximum required history length.



(a) Maximum observed history length.



(b) Minimum observed history length.



(c) Average observed history length.

Figure 7.13: The maximum (a), the minimum (b), and the average (c) observed required history length for SDM1 as a function of the Trellis order. Input stimuli consist of 1 kHz sine waves with amplitudes of -6 dB, -20 dB, -60 dB, and -100 dB.

Fig. 7.13(c) shows the average observed history length. The average required history length is approximately five times less than the maximum required length. The difference between the -6 dB, -20 dB, and -60 dB curves is relatively small and only for the -100 dB signal substantially more history length is required for making unambiguous decisions.

These experiments validate the expectation that a higher Trellis order (larger N), as well as a lower signal level, will require a larger Trellis depth L to unambiguously determine the output symbol. The impact of the Trellis order is larger than that of the signal level. On average the influence of the signal amplitude on the required history length is relat-

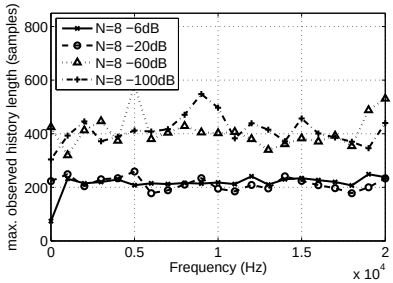
ively small, but occasionally more latency is required to find the output symbol, resulting in a maximum required latency which is approximately five times larger than the average required latency.

7.4.3 Trellis depth as a function of the signal frequency

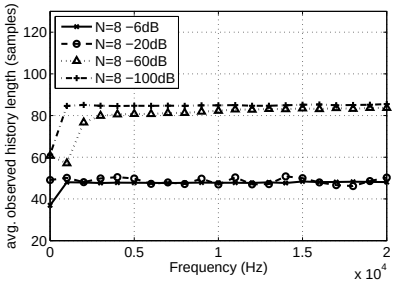
In ch. 6 it was reasoned that for baseband signal frequencies there should be no influence on the required Trellis depth, since the signal frequency is always low compared to the sampling rate. This assumption is verified by measuring the required Trellis depth for signal frequencies in the interval of 0 Hz to 20 kHz. The Trellis order is fixed at $N = 8$. The experiment is performed for SDM configuration SDM1 (see app. B) using different signal amplitudes (-6 dB, -20 dB, -60 dB, and -100 dB).

Fig. 7.14(a) shows that there is, as expected, no influence of the signal frequency on the maximum required history length. The signal amplitude does influence the required history length, which is in agreement with the results discussed above. A clear difference in the maximum observed length between the -6 dB and -20 dB inputs and the -60 dB and -100 dB input signals is visible.

The average observed history length is shown in fig. 7.14(b). Again, a clear difference in the average observed length between the -6 dB and -20 dB inputs and the -60 dB and -100 dB input signals is visible, but no influence of the signal frequency is present.



(a) Maximum required history length.



(b) Average required history length.

Figure 7.14: The maximum (a) and the average (b) observed required history length for SDM1 as a function of the input frequency for Trellis order $N = 8$. Input stimuli consist of sine waves with amplitudes of -6 dB, -20 dB, -60 dB, and -100 dB.

7.4.4 Trellis depth as a function of the loop-filter configuration

The influence of the loop-filter configuration on the required history length is investigated and compared between five loop-filter configurations (SDM1, SDM1FB, SDM2, SDM3, and SDM4 from app. B). For this comparison the Trellis order is varied from 1 to 10 and 1 kHz sine waves with several amplitudes are applied as input signals. In fig. 7.15 the obtained values of the average required history length are plotted for the different configurations for an input level of -60 dB. For other signal amplitudes similar results are obtained (not shown).

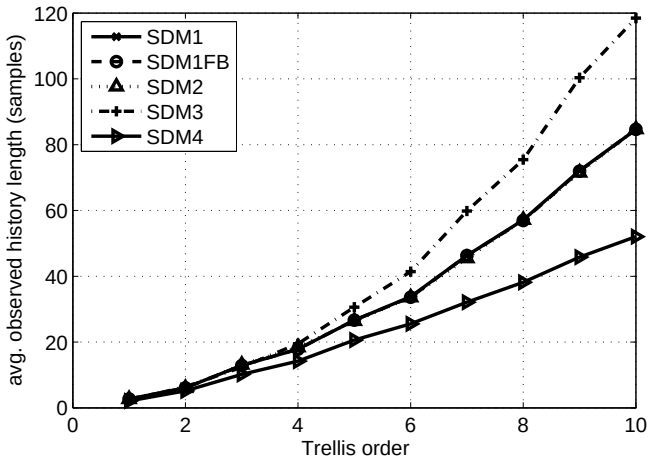


Figure 7.15: The average observed required history length for SDM1, SDM1FB, SDM2, SDM3, and SDM4 as a function of Trellis order for a 1 kHz-60 dB sine wave input. Note that the curves for SDM1, SDM1FB, and SDM2 are virtually overlapping.

In the figure the curves for configuration SDM1 and the feed-back variant SDM1FB can hardly be distinguished. Since the NTF of the two configurations is equal, and the STF in the baseband region is almost equal, the required history length is also equal. Configuration SDM2 is very similar to SDM1, i.e. the difference is the addition of two resonator sections, and also here the required history length is virtually equal to the results for SDM1. Apparently the presence of resonator sections does hardly influence the required history length. This result puts question marks to the hypothesis of Angus [3] that the presence of

resonators increases the required history length.

The observed required history length values for the aggressive filter configuration SDM3, i.e. a fifth order loop-filter with a corner frequency of 140 kHz and two resonator sections, are higher than those of SDM1 and SDM2. The percentage-wise difference in average observed history length increases as a function of the Trellis order. At $N = 10$ the average observed history length is approximately 40% higher for SDM3, indicating that more solutions with near equal quality exist for the aggressive loop filter. The third order noise shaping of SDM4 results in a shorter observed history length than SDM1. This result is in agreement with the observation that more aggressive noise shaping results in longer history lengths.

In ch. 6 it was postulated that aggressive loop filters will require more latency before a unique solution is found, but only for low input levels. For high input levels it was expected that the required history length for an aggressive loop filter would reduce, since only a limited amount of feed-back sequences can generate such a signal. However, in the experiments this behavior was not detected, i.e. also for high input levels the average required history length was larger for the aggressive loop-filter configuration. Apparently, since only during a fraction of the sine wave period the amplitude is close to the extreme values, the influence on the average required history length is very limited. Thus, a more aggressive loop filter, when realized with a higher filter corner frequency, will typically require a larger latency than a mild loop filter. The addition of resonator sections which also increases the SNR of the system does hardly influence the required average history length. The reason for this discrepancy is unclear.

7.4.5 Summary

By means of simulations the hypotheses from ch. 6 on how the Trellis order, the input signal amplitude and frequency, and the loop-filter configuration influence the required Trellis depth for unambiguously determining the output symbol have been verified. It has been found that the required history length increases slightly faster than linear as a function of the Trellis order. Low amplitude signals require approximately twice the history length than high amplitude signals. As expected, the signal frequency has no influence on the required latency.

Aggressive filter configurations, i.e. higher loop-filter corner frequencies or higher order filters, result in longer history lengths while less aggressive filter configurations result in shorter history lengths. Surprisingly,

the presence of resonators in the loop filter does not influence the required history length. There is no difference in the required history length between a feed-forward loop filter and the equivalent feed-back filter.

From the simulations it has been found that, typically, the maximum observed required history length is 5 times the average observed history length. However, situations in which the maximum is 8 times higher than the average have been observed. For the studied loop-filter configurations a history length of 1000 bits is sufficient for Trellis order $N = 10$ to avoid truncation noise, while for the case $N = 4$ only a length of 200 bits is required. However, since a longer history length has a negative impact on the computational performance and requires more hardware resources, it is worthwhile to perform simulations for a given combination of the Trellis order and loop-filter configuration to determine the required history length when realizing an optimized design.

7.5 Functional performance

The signal conversion performance that can be realized by a Trellis SDM is only minimally documented by Kato in [38]. In order to get a complete picture, the functional performance of a Trellis SDM is investigated on the basis of the classical signal quality indicators SNR, SINAD, THD, and SFDR. Next to these indicators, the SDM specific performance measures stability and noise modulation are investigated. Finally, a summary is given of the Trellis SDM functional performance.

7.5.1 SNR, SINAD, THD and SFDR

By means of simulations the signal conversion performance of the Trellis algorithm is investigated and compared to the performance of a traditional SDM. All experiments are performed for a 1 kHz -6 dB input signal, for Trellis orders of 1 to 8. In order to always guarantee convergence the Trellis depth is set to $L = 2048$. Several loop-filter configurations are used.

The FFT length is 1 million samples and 4 power averages are applied. In order to get an accurate measure of the THD, 128 coherent[1] averages are

[1] Coherent averages are performed on the time domain signal before calculating the power spectrum and highlight coherent signal components, while power averages are performed on the power spectrum to smoothen the spectrum. See app. A for a detailed explanation.

performed before the power averages are made. The first 9 harmonics that fall in the 0-20 kHz band are taken into account for the THD measurement.

SDM1

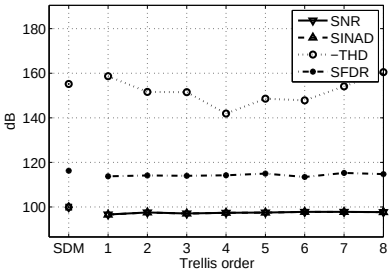
In fig. 7.16(a) the SNR, SINAD, THD, and SFDR are plotted for loop-filter configuration SDM1 (see app. B) for both a normal SDM and a Trellis modulator.

As expected, the SINAD and SNR are not influenced by the Trellis order, and are constant at a level of 98 dB. Since the SINAD is equal to the SNR there are no distortion components above the noise-floor. As a result, the SFDR level is set by non-harmonic components and is therefore constant as well. A comparison of the Trellis SDM performance with that of the normal SDM reveals that the SNR and SINAD of the normal SDM are slightly higher, which is in agreement with the predictions based on the linear model. Since the SFDR is limited by the quantization noise, this figure is also slightly higher for the normal SDM.

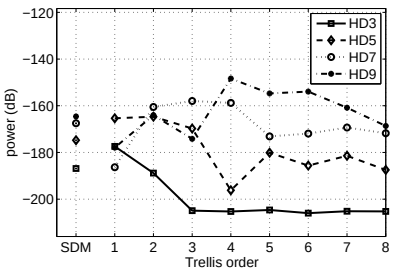
The THD behavior is not very consistent, i.e. first the THD increases to reach a maximum for $N = 4$, after which it again decreases to its initial value. This behavior can be understood by looking at the power in the individual harmonics. In fig. 7.16(b) the power in the first four odd harmonics is plotted (no even harmonics are present in the output). When the Trellis order increases, the power in HD3 and HD5 decreases, while the higher order harmonics first become stronger before they also start to reduce. The largest harmonic is realized for $N = 4$, matching the point of minimum THD. Comparison of the power distribution with that of the normal SDM shows that for large N the results are similar, but that for small N the power distribution is very different. The level of the harmonics is nearly equal to that of the normal SDM, resulting in a similar THD value.

SDM1FB

For loop-filter configuration SDM1FB (see app. B), the feed-back version of SDM1, the SNR, SINAD, THD, and SFDR are plotted in fig. 7.17(a). As expected, the performance is very similar to that of the feed-forward configuration. However, a close inspection of the harmonics, shown in fig. 7.17(b), reveals that there is a difference between the performance of the feed-forward and feed-back filter. For both the normal SDM and



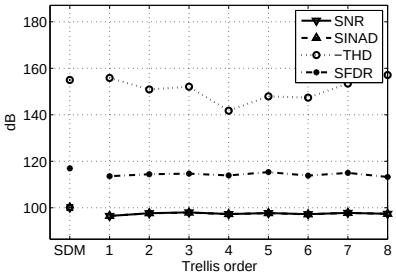
(a) SNR, SINAD, THD, and SFDR.



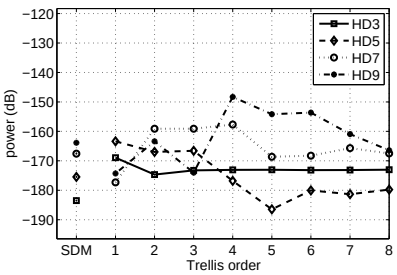
(b) Power in odd harmonics.

Figure 7.16: SNR, SINAD, THD, and SFDR performance (a) and the power in HD3, HD5, HD7, and HD9 (b) for SDM1 as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB. Normal SDM performance is shown for reference.

the Trellis modulator the power in the third harmonic is higher for the feed-back filter than for the feed-forward filter. More specifically, the strong reduction in HD3 realized for the feed-forward configuration is not present in the case of the feed-back filter. The behavior of the other harmonics is very similar. Since the quantization noise at the end of the pass-band is at a higher level than the harmonics, the higher HD3 harmonic level has no impact on the SNR, SINAD, and SFDR.



(a) SNR, SINAD, THD, and SFDR.



(b) Power in odd harmonics.

Figure 7.17: SNR, SINAD, THD, and SFDR performance (a) and the power in HD3, HD5, HD7, and HD9 (b) for SDM1FB as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB. Normal SDM performance is shown for reference.

SDM2 and SDM2FB

For loop-filter configuration SDM2 (see app. B) the SNR, SINAD, THD, and SFDR are plotted in fig. 7.18(a). For N below 3 the SNR is slightly higher than the SINAD, whereas for higher Trellis orders the SNR and SINAD are equal and constant at 114 dB, indicating that all harmonics are below the noise floor. In all cases the SINAD and SNR of the normal SDM are approximately 2 dB higher than those of the Trellis SDM. The SFDR increases from 118 dB for $N = 1$ to 137 dB for $N = 5$ and above. The THD decreases from -116 dB for $N = 1$ to -140 dB for $N = 8$. For $N = 3$ the THD and SFDR of the Trellis modulator are comparable to that of the SDM. For higher Trellis orders the THD and SFDR of the Trellis modulator are better.

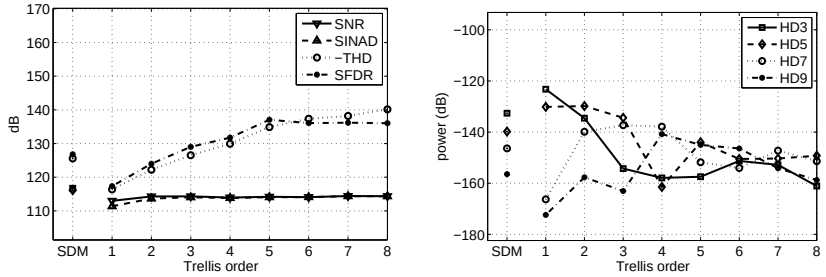
Fig. 7.18(b) shows the power in HD3-9 for configuration SDM2. An increase of the Trellis order does not result in a monotonous decrease of each of the individual harmonics, but does result in a decrease of the largest harmonic. For N larger than 2 the largest harmonic of the Trellis modulator is already below the largest harmonic of the normal SDM. Increasing the Trellis order is clearly effective for this loop-filter configuration to reduce harmonic distortion.

The performance of configuration SDM2FB (see app. B), the feedback version of SDM2, is virtually identical to that of SDM2. The only noticeable difference, for both the normal SDM and the Trellis SDM, is that the performance of the feed-back configuration is approximately 2 dB lower than that of the feed-forward modulator. Since the behavior of the harmonic distortion components is equal as well no graphs are shown.

SDM3 and SDM3FB

For loop-filter configuration SDM3 (see app. B) the SNR, SINAD, THD, and SFDR are plotted in fig. 7.19(a). No performance is reported for the reference SDM because the modulator becomes unstable with the test signal. The Trellis modulator is also unstable for $N = 1$. For all N above 1, the SNR and SINAD equal 123 dB. The SFDR increases from 141 dB to 145 dB when N increases from 2 to 4. Because the SFDR is already very large for $N = 2$, there is no noticeable influence on the SNR for larger N. The THD decreases from -139 dB for $N = 2$ to -162 dB for $N = 8$. For N above 2 harmonic components are not limiting the SFDR.

Fig. 7.19(b) shows the power in the first four odd harmonics. A near

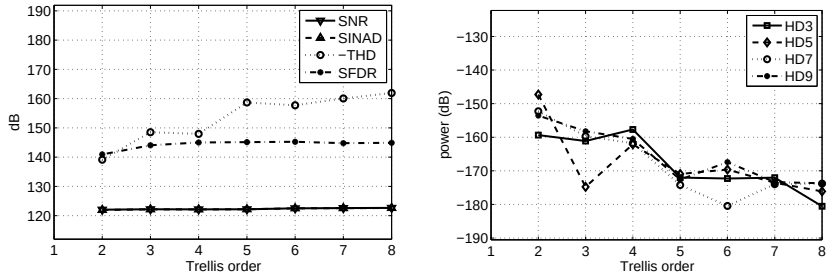


(a) SNR, SINAD, THD, and SFDR. (b) Power in odd harmonics.

Figure 7.18: SNR, SINAD, THD, and SFDR performance (a) and the power in HD3, HD5, HD7, and HD9 (b) for SDM2 as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB. Normal SDM performance is shown for reference.

monotonous decrease in power is realized for every harmonic, resulting in a decrease from -147 dB at $N = 1$ to -173 dB for $N = 8$ for the largest harmonic component.

Similar to the situation above, the performance of the feed-back configuration is nearly identical to that of the feed-forward configuration. In this case the SNR and SINAD are approximately 1 dB lower for the feed-back filter. The behavior of the harmonics is similar as well.



(a) SNR, SINAD, THD, and SFDR. (b) Power in odd harmonics.

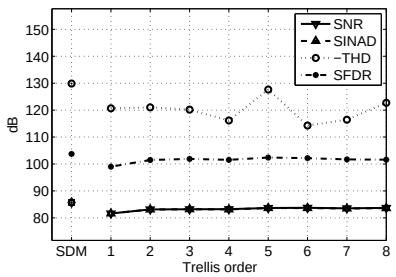
Figure 7.19: SNR, SINAD, THD, and SFDR performance (a) and the power in HD3, HD5, HD7, and HD9 (b) for SDM3 as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB. No normal SDM reference point shown because of instability.

SDM4 and SDM4FB

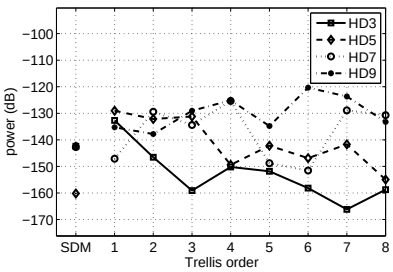
For loop-filter configuration SDM4 (see app. B) the SNR, SINAD, THD, and SFDR are plotted in fig. 7.20(a). The behavior for this configuration shows a strong resemblance to that of SDM1. The SNR equals the SINAD for every Trellis depth. For $N = 1$ the SNR is 82 dB, while for larger N the SNR is constant at 83 dB, which is approximately 2 dB lower than realized by the normal SDM. The SFDR shows a similar behavior, i.e. for N above 1 the SFDR is constant at 101 dB while for $N = 1$ the SFDR is 99 dB. The THD value is relatively constant for all values of N, with the exception of a (random) peak for $N = 5$.

The power in the first 4 odd harmonics is plotted in fig. 7.20(b). As expected, the behavior is very similar to that of SDM1. More specifically, for a larger N the power in HD3 and HD5 decreases, while the higher order harmonics stay more or less on the same level. In this case a larger N does not result in a smaller maximum harmonic.

The results (not shown) for the feed-back configuration are identical to that of the feed-forward configuration.



(a) SNR, SINAD, THD, and SFDR.



(b) Power in odd harmonics.

Figure 7.20: SNR, SINAD, THD, and SFDR performance (a) and the power in HD3, HD5, HD7, and HD9 (b) for SDM4 as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB. Normal SDM performance is shown for reference.

Summary

From the signal conversion experiments it can be concluded that a higher Trellis order typically results in less harmonic distortion in the output signal. For the loop filters without resonator sections (SDM1, SDM4) the

level of the first 9 harmonics is typically already very low in the normal SDM case, and no real improvement can be realized by using a high Trellis order since the harmonics are not limiting performance. When resonator sections are present, signal harmonics are limiting the THD of a normal SDM, and the THD can be improved by 10 to 20 dB compared to a normal SDM by using a Trellis order $N = 8$. Since a higher Trellis order does not result in an increase in SINAD, the improvement in SFDR is often less. In the typical SDM case where harmonics are causing the SINAD value to be lower than the SNR, a Trellis SDM of a low Trellis order can already improve the SINAD to the same level as the SNR. However, when a Trellis SDM is used instead of a normal SDM, the SNR will be approximately 2 dB lower if the same loop filter is used and both converters are undithered. However, the normal SDM will require dithering to reduce the harmonic distortion in the output and break up limit cycles and idle tones, reducing its SNR. The Trellis SDM, on the other hand, does not require dithering if a moderate Trellis order is used. As a result, in practice the difference in SNR between the two solutions will be less. Furthermore, application of a more aggressive loop filter can restore the SNR of the Trellis SDM to the level of the SDM (see sec. 7.5.2). When Trellis feed-forward and feed-back filter configurations are compared, hardly any difference can be detected. More specifically, both filter structures react in the same way to the Trellis algorithm and show very similar performance improvements.

7.5.2 Converter stability

According to the theory developed in ch. 6, the stability of a Trellis SDM should increase with the Trellis order. This effect is studied by means of simulations. The stability is investigated in two ways, namely by measuring the maximum input amplitude that can be converted without causing instability, and by measuring the highest filter corner frequency that can be used to convert a -6 dB input signal. All the experiments are performed for Trellis orders 1 up to 10 and for a normal reference SDM. In order to get accurate results an FFT length of $1 \cdot 10^6$ samples is used in combination with 4 power averages.

Maximum stable input amplitude

The stability of a Trellis SDM is measured by determining the maximum input level of a 1 kHz sine wave that can be converted without causing instability to the modulator. Instability is detected by comparing the

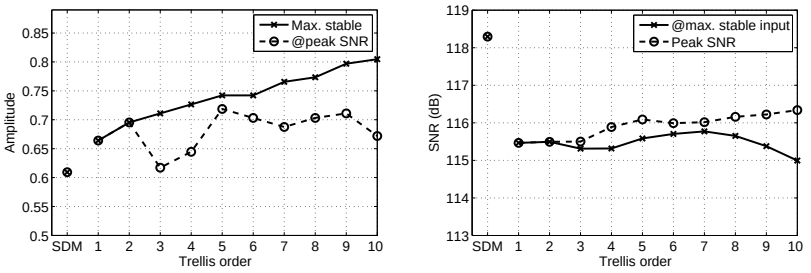
realized output SNR with the expected SNR, i.e. if a large discrepancy is found the converter is unstable. The input level just before instability occurs, i.e. the maximum stable input level, is recorded as well as the SNR for this input level. Since the peak SNR of an SDM is typically realized for a smaller input level, this point is also determined and recorded. The experiment is performed for two different modulator configurations, namely configuration SDM2, a fifth order modulator with resonators, and configuration SDM4, a simple third order modulator (see app. B for details).

SDM2 Fig. 7.21 shows the amplitude stability results obtained for configuration SDM2. In fig. 7.21(a) the maximum stable input amplitude that was found for the various Trellis orders is plotted, as well as the input amplitude that resulted in the maximum SNR. The associated SNR values are reported in fig. 7.21(b).

For the Trellis SDM, the maximum stable amplitude increases from 0.67 for $N = 1$ to 0.81 for $N = 10$, while the normal SDM is only stable for inputs up to 0.62. Unexpectedly, the SNR realized at the maximum input amplitude is nearly constant at 115.5 dB, and even decreases slightly for the higher Trellis orders. Since the amplitude increases by 20% from $N = 1$ to $N = 10$, an SNR increase of 1.6 dB would be expected for a linear system. In the case of the normal SDM the peak SNR (118.4 dB) is reached for the maximum input amplitude. For N larger than four, the Trellis SDM reaches a maximum SNR for an input of approximately 0.70, independently of the Trellis order. The SNR at this point is approximately 116 dB, and is, as expected, independent of the Trellis order. For the case of $N = 3$ and $N = 4$ the peak SNR is realized for smaller input levels than for the case of $N = 1$ and $N = 2$, although the SNR values for $N = 3$ and $N = 4$ are higher. Why the peak SNR is realized for significantly lower amplitudes is unclear.

SDM4 The amplitude stability results obtained for configuration SDM4 are depicted in fig. 7.21. Fig. 7.22(a) shows that also for the 3rd order modulator the maximum stable input that can be applied increases as a function of the Trellis order. As a reference point, the maximum stable input for a normal SDM is added to the figure. For $N = 1$ the maximum stable input is 0.82 while the SDM is only stable up to 0.73. For $N = 10$ a maximum input of 0.92 can be applied. The amplitude that results in the peak SNR is approximately 0.65, independent of the Trellis order.

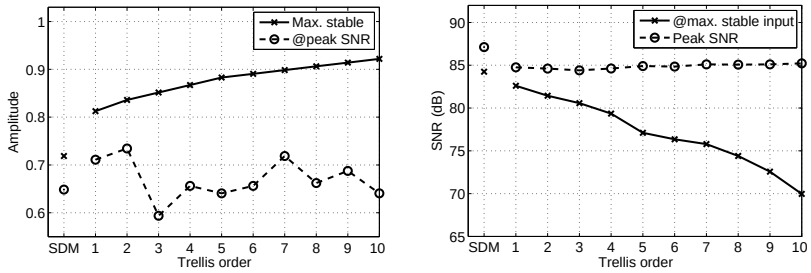
In fig. 7.22(b) the SNR at the maximum stable input is plotted, as well as the peak SNR. The plot shows a strong reduction of the SNR at the



(a) Stable input amplitude. (b) SNR.

Figure 7.21: Maximum stable input amplitude and input amplitude for peak SNR (a) and the associated SNR (b) as a function of the Trellis order for configuration SDM2 for a 1 kHz sine wave. Normal SDM performance is shown for reference.

maximum stable input as a function of the Trellis order. This behavior is different from that of the 5th order modulator, that shows a nearly constant SNR at the maximum input level. The peak SNR is constant at 85 dB, as expected.



(a) Stable input amplitude. (b) SNR.

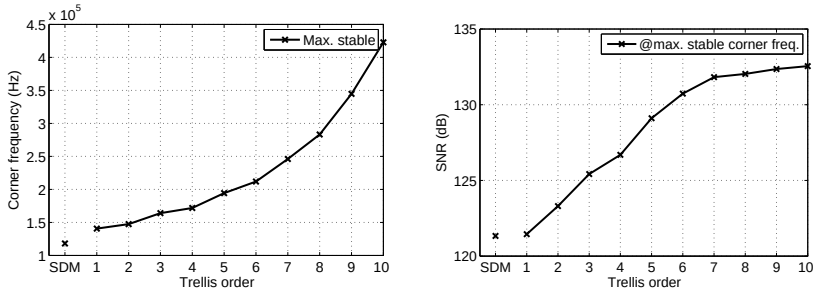
Figure 7.22: Maximum stable input amplitude and input amplitude for peak SNR (a) and the associated SNR (b) as a function of the Trellis order for configuration SDM4 for a 1 kHz sine wave. Normal SDM performance is shown for reference.

Maximum loop-filter corner frequency

The stability of a Trellis SDM is also measured by determining the maximum loop-filter corner frequency that can be used to convert a -6 dB 1 kHz sine wave without causing instability to the modulator. Instability is detected by comparing the realized output SNR with the expected SNR, i.e. if a large discrepancy is found the converter is unstable. The corner frequency just before instability occurs, i.e. the maximum stable corner frequency, is recorded as well as the SNR at this point. The experiment is performed for a loop filter that is based on configuration SDM2, and for a loop filter that is based on configuration SDM4 (see app. B for details).

SDM2 In the experiment the corner frequency of a loop filter similar to configuration SDM2, i.e. a fifth order Butterworth design with two resonators sections, is varied until the point of instability is found. The results of this experiment are depicted in fig. 7.23. In fig. 7.23(a) the maximum loop-filter corner frequency that results in stable conversion is plotted as a function of the Trellis order. As expected, a higher filter corner frequency can be used when the Trellis order increases. The SNR that is realized at this maximum corner frequency is plotted in fig. 7.23(b). The SNR that is realized by the normal SDM at its highest possible corner frequency is virtually equal to that of the Trellis SDM with $N = 1$, although the corner frequency of the Trellis SDM is much higher. The increasingly higher corner frequencies for larger Trellis orders result in an increasingly higher SNR. However, from $N = 7$ onwards, i.e. for corner frequencies of 250 kHz and higher, the SNR is nearly constant. This phenomenon of a saturating SNR is present for all look-ahead modulators, and will be studied in more detail in chapter 12.

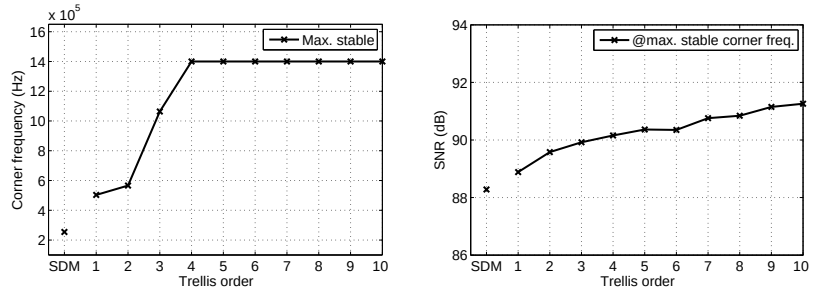
SDM4 The corner frequency of a loop filter similar to configuration SDM4, i.e. a third order Butterworth design, is varied until the point of instability is found. The results of this experiment are depicted in fig. 7.24. In fig. 7.24(a) the maximum loop-filter corner frequency that results in stable conversion is plotted as a function of the Trellis order. From $N = 4$ onwards the maximum corner frequency of the filter is equal to half the sampling rate, i.e. the maximum corner frequency possible. Still, the SNR increases slightly at this point when the Trellis order is increased (fig. 7.24(b)) from $N = 4$ to $N = 10$. This result is unexpected, since for less extreme situations a higher Trellis order does not improve the SNR (see sec. 7.5.1). At the maximum possible corner frequency of



(a) Stable corner frequency. (b) SNR.

Figure 7.23: Maximum loop-filter corner frequency that results in stable operation as a function of the Trellis order for a -6 dB sine wave with filter configuration SDM2 (a) and the associated SNR (b). Normal SDM performance is shown for reference.

the SDM its SNR is slightly lower than that of the Trellis SDM with $N = 1$.



(a) Stable corner frequency. (b) SNR.

Figure 7.24: Maximum loop-filter corner frequency that results in stable operation as a function of the Trellis order for a -6 dB 1 kHz sine wave with filter configuration SDM4 (a) and the associated SNR (b). Normal SDM performance is shown for reference.

Summary

As predicted in ch. 6, a Trellis converter becomes more stable when the Trellis order is increased. The increase in stability has been verified by

measuring the maximum signal amplitude that can be converted without causing instability to the converter, and by measuring what maximum filter corner frequency can be used for a given signal level. Both experiments show a significant improvement in stability, independent of the filter configuration.

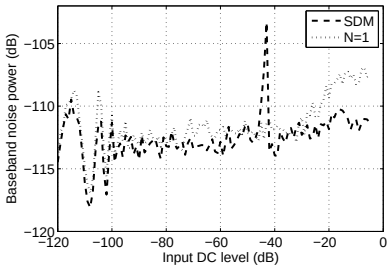
More specifically, the stable input range can be increased by 15 to 20% by using Trellis order $N = 10$ instead of $N = 1$. However, the amplitude that results in the peak SNR is not affected by the increase in Trellis order. Therefore, the increase in input range does not result in an improvement of the SNR of the converter. Still, the increase in input range can be very useful if it is desirable to be able to handle large input signals, for example, in the case of SA-CD mastering.

If the increase in stability is used to stabilize a more aggressive filter, i.e. a loop filter with a higher corner frequency, an SNR improvement can be realized. An SDM at its most aggressive setting and a Trellis SDM with $N = 1$ at its most aggressive setting realize virtually the same SNR. By increasing the Trellis order N a more aggressive filter can be used, and a higher SNR is, typically, resulting. For example, the experiment with prototype filter SDM2 shows that an improvement of 10 dB can be realized by using $N = 7$ instead of $N = 1$. However, in the case of the experiment with SDM4, a third order filter, only an improvement of 2 dB was realized. Here, the theoretical maximum filter corner frequency is reached for $N = 4$, and it is not possible to design a more aggressive third order filter, limiting the improvements. In this case the filter order would need to be increased to four in order to enable more aggressive noise shaping, which is required to obtain a higher SNR.

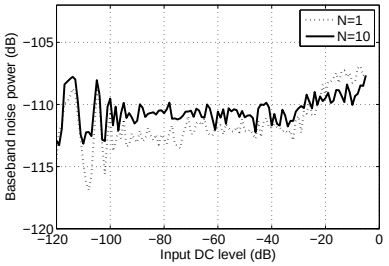
7.5.3 Noise modulation

The in-band noise modulation of a Trellis SDM is investigated by sweeping a DC input signal and measuring the amount of noise present in the 0-20 kHz band. The experiment is performed for loop-filter configuration SDM1 (see app. B), realized as a Trellis SDM with $N = 1$ and $N = 10$, and also as a normal SDM as a reference. Two experiments are performed, i.e. once with a logarithmic selection of the DC levels and once with a linear selection. The logarithmic selection is useful to illustrate how the amount in-band quantization noise varies globally as a function of the amplitude of the input signal. The linear amplitude selection criterion results in rational DC levels, which will often trigger limit cycles in a typical SDM.

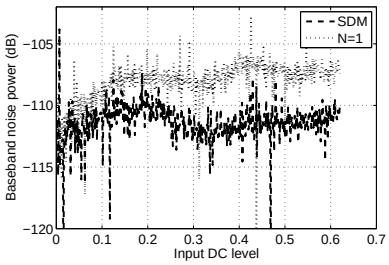
In the first experiment a sweep is performed for input levels of -120 dB to



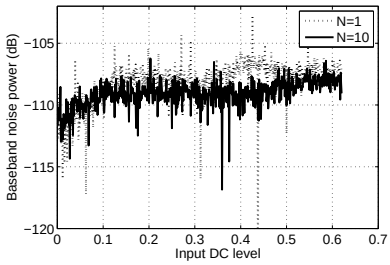
(a) SDM vs. TSDM $N = 1$ (log sweep)



(b) TSDM $N = 1$ vs $N = 10$ (log sweep)



(c) SDM vs. TSDM $N = 1$ (linear sweep)



(d) TSDM $N = 1$ vs $N = 10$ (linear sweep)

Figure 7.25: In-band noise as a function of the DC input level. Comparison of an SDM with a Trellis SDM with $N = 1$ in fig. (a) for a logarithmic input sweep and in fig. (c) for a linear input sweep. Comparison of a Trellis SDM with $N = 1$ and $N = 10$ in fig. (b) and fig. (d) for a logarithmic, respectively, linear input sweep.

-5 dB in steps of 1 dB. The results for the normal SDM and the Trellis SDM with $N = 1$ are plotted in fig. 7.25(a). For small input levels both converters produce the same amount of in-band noise. However, for large DC levels the Trellis SDM generates up to 6 dB more noise, resulting in a lower output SNR. The SDM, on the other hand, shows a large peak in the in-band noise for an input level around -44 dB.

In fig. 7.25(b) the Trellis SDM with $N = 1$ is compared with the Trellis SDM with $N = 10$. For low level input signals the Trellis SDM with $N = 10$ produces more in-band noise than the converter with $N = 1$. However, at large DC levels the noise increases only minimally for the $N = 10$ converter whereas the $N = 1$ converter shows a strong increase in the in-band noise. Thus, the variation in noise is less for the $N = 10$ Trellis SDM converter. Comparison of $N = 10$ with the SDM in

fig. 7.25(a) shows that the amount of variation in noise is approximately the same for the two, except that the SDM has a peak in the in-band noise for inputs around -44 dB.

For normal Sigma-Delta Modulators it is known that certain specific DC input levels can result in a strong reduction or increase of the in-band noise. At these specific levels, typically rational DC levels, the modulator changes its behavior and will generate a limit cycle, i.e. it will generate an output spectrum that consists of discrete tones only. If the limit cycle only contains high frequency components the in-band noise will be zero, but also limit cycles exist that have a significant in-band component. In order to test if also Trellis Sigma-Delta Modulators can exhibit this behavior, a second experiment is performed in which the input level is varied in steps of $\frac{1}{1024}$.

The results are plotted in fig. 7.25(c) for the normal SDM and the Trellis SDM with $N = 1$, and in fig. 7.25(d) for the case of $N = 1$ and $N = 10$. As predicted, there are certain DC levels for which the SDM does not produce any, or very little, in-band noise. The behavior of the Trellis SDM with $N = 1$ is very similar to that of the SDM, i.e. also here there are levels that result in little or no in-band noise. The DC level for which the SDM produces a high frequency limit cycle are different from the Trellis SDM levels, but this is only to be expected since the noise-shaping characteristic of the two converters is different. Both converters also have a relatively large number of DC levels that cause an increase in the in-band noise.

Comparison of the Trellis SDM with $N = 1$ and the Trellis SDM with $N = 10$ in fig. 7.25(d) reveals that for the $N = 10$ converter no levels where triggered that result in zero in-band noise. This is a surprising result since both converters have the same loop filter and realize the same noise-shaping characteristic. Furthermore, if no in-band noise is generated the in-band match of the quantized signal with the input signal is perfect, and it could be expected that such a solution would be preferred over alternatives with in-band noise. However, apparently the tones at high frequencies cause an error signal which is larger than that of a longer alternative solution signal with in-band noise. As a result, the amount of noise generated by the Trellis SDM with $N = 10$ is more constant than by the converter with $N = 1$ or the SDM, with fewer DC levels that result in an increase or decrease of the noise.

Although the described experiments are not exhaustive and are no proof, they make it plausible that application of a Trellis SDM instead of a normal SDM can reduce the severeness of noise modulation. For high quality audio applications this reduction in the variation of the in-band

noise is a benefit of the Trellis SDM over the normal SDM. However, the experiments also show that even in the case of a Trellis SDM with $N = 10$ there still remains some noise modulation. Thus, a Trellis SDM is not able to solve the problem of noise modulation completely.

7.5.4 Summary

It has been shown that the SNR that is realized by a Trellis SDM is, in typical conditions, independent of the Trellis order. Only in the special situation where the corner frequency of the filter is equal to the maximum frequency possible a higher SNR is realized when a higher Trellis order is used. The SINAD on the other hand, is for practical loop filters a function of the Trellis order. More specifically, an increase in the Trellis order reduces the THD of a converter by a larger amount, and the SINAD eventually becomes equal to the SNR. Improvements of the THD up to 20 dB for $N = 10$ have been demonstrated for loop filters with resonator sections. With such loop filters the SFDR is typically limited by the harmonic distortion, and application of a Trellis SDM with N of four or larger reduces the distortion components to the noise-floor level. Loop filters without resonator sections show a different behavior, since here the distortion components are typically already much lower than the noise-floor at the end of the pass-band, and the SFDR can therefore not be improved. A comparison of the SNR achieved by a Trellis SDM with the SNR achieved by a normal SDM with the same loop filter and both converters without dithering shows that the normal SDM, typically, achieves a 2-3$\tilde{\text{d}}$B better result. This observation is in line with predictions based on the linear NTF model. However, a normal SDM will require some dithering in order to limit harmonic distortion and to avoid limit cycles and idle tones, reducing its SNR, whereas a Trellis SDM does not require any dithering if a moderate Trellis order is used. Therefore, in practice the difference in SNR between the two converters will be very limited.

Despite the disadvantage of a theoretically reduced SNR, the Trellis algorithm offers significant advantages. Application of a large Trellis order improves the stability of the converter significantly. For example, the input range can be improved by 10 to 20%, depending on the loop-filter configuration. Although an increase of the stable input range does not or hardly improve the maximum SNR of the converter, such an increase can be very beneficial for, for example, SA-CD mastering applications. Alternatively, it is possible to increase the corner frequency of the loop filter without reducing the stable input range. Such an increase in the

corner frequency improves the noise-shaping effectiveness and can more than compensate for the reduced SNR.

The typical SDM problem of in-band noise modulation can only be reduced, but not removed, by the use of a Trellis modulator. Experiments show that for a larger Trellis order there are fewer DC levels that cause idle tone behavior and that the amount of noise variation is less, but even with $N = 10$ the amount of baseband noise is not constant.

7.6 Implementation aspects

By carefully implementing the algorithm steps described in sec. 7.1 (Kato model) or in sec. 7.2 (pruned look-ahead model), a working modulator can be realized. However, since the algorithm requires a significant amount of computational resources, it is worthwhile to investigate the implementation challenges, such that an efficient implementation can be realized.

7.6.1 Required computational resources

Nearly all the resources required for the Trellis algorithm scale with the Trellis order N. For a look-ahead order of N, 2^N parallel look-ahead filter units are required. Each processing unit requires a memory of L bits to store the path history, resulting in a total of $2^N \cdot L$ bits of memory. The value of L is a function of N (sec. 7.4), therefore the amount of memory more than doubles if the look-ahead depth is increased by one. The two possible output values of each processing unit, i.e. the path cost for continuing with a '+1' and for a '-1', are passed to the evaluate and select function, which selects the 2^N solutions that continue. This selection requires the addition of the two path cost values to the running path cost, and the comparison of 2^N times two values. Finally, there is the Viterbi trace-back operation if the algorithm is implemented as proposed by Kato. If an implementation according to the generic look-ahead framework is made this resource is not required, but in return more memory access is required.

7.6.2 Look-ahead filter unit

For an efficient implementation of the Trellis sigma-delta modulation algorithm it is necessary to have an optimized look-ahead processing

unit. These optimizations relate to the latency of the processing unit, resource sharing, and application of potential dither signals.

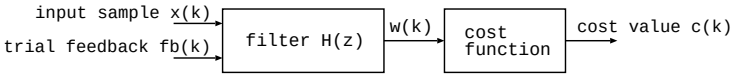


Figure 7.26: Generic Trellis SDM look-ahead filter structure.

Latency

In the first step of the Trellis sigma-delta modulation algorithm the cost for both trial output candidates is calculated. This calculation is accomplished by applying an input sample and a trial feedback symbol (either '+1' or '-1') to the generic look-ahead filter structure, depicted in fig. 7.26. The output of the structure is the cost value. If no direct path, i.e. no path with zero clock cycles delay, from the trial feedback input to the cost value output exists, the influence of the input symbol on the output can only be assessed in a later clock cycle. As a result, independent of the feedback value applied, the same cost value will be presented at the output. If no special care is taken to clock the structure such that the output signal is updated and is reflecting the influence of the input signal, an incorrect conclusion on the quality of the feedback signal will be drawn. Therefore, the filter $H(z)$ should have a direct path from the input to the output such that the influence of the feedback signal can be assessed directly.

A typical SDM feed-forward loop filter, depicted in fig. 7.27(left), does not have a direct path from input to output, but can be trivially adapted, as shown in the right half of the same figure. Note that the coefficients are moved to the input of the integrator registers in the Trellis SDM structure. The internal state values of the two structures are identical, but the output signals are offset by one clock cycle.

In the case of a feed-back filter the SDM filter, shown in the left half of fig. 7.28, the filter can be adapted to a Trellis SDM filter by changing the output node, as shown in the right half of the figure. Also in this case the internal state values of the two filter structures are identical.

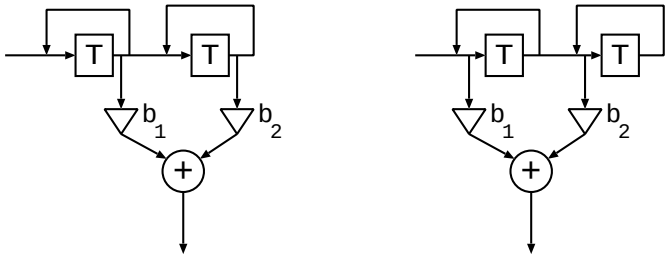


Figure 7.27: Typical second order feed-forward SDM loop filter (left) and corresponding Trellis SDM loop filter (right).

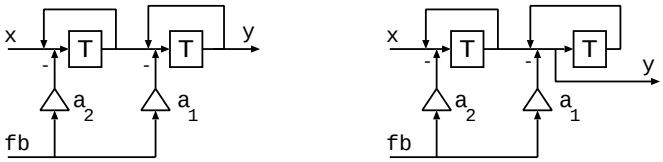


Figure 7.28: Second order feed-back Sigma-Delta filter (left) and Trellis SDM equivalent (right).

Resource sharing

Every Trellis SDM look-ahead filter unit has to calculate the cost for appending the bitstream with a '+1' and '-1' symbol. The straightforward approach is to apply both feed-back values in combination with the main input sample subsequently to the loop filter. However, a large number of computations can be saved by re-using results from earlier computations.

For example, the part of the loop-filter output that is resulting from the filter state does only need to be calculated once. If the output of the filter is w_{+1} for a feed-back value of '+1', the output of the filter for the feed-back value of '-1' will equal

$$w_{-1} = w_{+1} - 2 \cdot a_1 \tag{7.7}$$

This can be easily seen if it is recognized that

$$w_{+1} = w_0 + 1 \cdot a_1 \tag{7.8}$$

where w_0 is the filter output that is obtained for a zero input signal, and

that

$$\begin{aligned} w_{-1} &= w_0 - 1 \cdot a_1 \\ &= w_{+1} - 2 \cdot a_1 \end{aligned} \tag{7.9}$$

In a similar fashion, some of the computations required to update the internal states of the filter can be re-used. For example, in the case of a feed-forward filter, only the state of the first integrator is dependent on the value of the feed-back value. Thus, if the path is extended with both a '+1' and a '-1' symbol, only the state of the first integrator needs to be calculated twice. In the case of a feed-back modulator the internal states are all dependent on the feed-back value. Still, also here half of the computations can be shared between the two feed-back symbols, i.e. one addition and one subtraction are required to update an integrator but the addition is not dependent on the feed-back symbol.

Dither

In a normal SDM, typically, a small amount of dither is added to the quantizer input to reduce the harmonic distortion and to break idle tones and limit cycles. From sec. 7.5 it is clear that there is little reason for dithering a Trellis SDM, since with a moderate Trellis order the harmonic distortion is already suppressed significantly. However, noise-modulation can not be fully removed by using a Trellis SDM, and only with a very high Trellis order the occurrence of limit cycles reduces. Therefore, it can still be desired to dither the Trellis SDM mildly.

In a Trellis SDM there are two basic locations for adding dither that are comparable to the way dither is typically added in a normal SDM. The first location is at the output of the loop filter, which would be the input to the quantizer in a normal SDM. The second location for adding dither is after application of the cost function. The two possible dither locations are indicated in fig. 7.29 by dither signals *dither*1 and *dither*2.

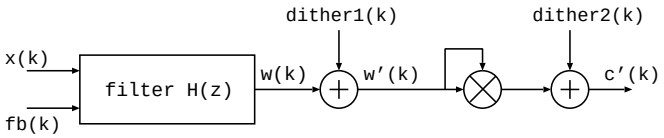


Figure 7.29: Possible locations for adding dither in a Trellis SDM indicated by signals *dither*1 and *dither*2.

From the two dither locations the second is preferred over the first, since here there will be no correlated component between the dither signal and the filter output added to the final cost score. If dither is added at the first location there is the possibility that the spectrum of the dither is shaped by the signal, potentially introducing coding artifacts. At the second location dither with any probability density function (PDF), including spectrally shaped dither, can be applied.

The generation of good quality digital dither is a computational intensive task. In the Trellis sigma-delta modulation algorithm there are 2^{N+1} cost values calculated, all requiring a dither value. Unfortunately it is not possible to use the same dither value for all the paths, since this does not change the path values relatively to each other, resulting in a zero nett effect. However, since all these computations will result in a single solution the dither values can be correlated. More specifically, if the dither value that is used for, for example, the first path in the first clock cycle is used for the second path in the second clock cycle, the converter is properly dithered. Thus, by re-using the dither values 2^N times a significant saving in computations can be realized at the cost of a small additional memory.

7.6.3 Output symbol selection

While all the operations of the Trellis sigma-delta modulation algorithm can be executed in parallel, the output symbol selection process is a sequential one. More specifically, if the output symbol selection is performed as proposed by Kato, in order to trace back L clock cycles L times a memory index needs to be determined which depends on the previously selected index. Such an operation is inefficient in both a hardware and a software realization. Furthermore, if the Viterbi algorithm is used to determine the output symbol it is not possible to easily detect if the algorithm has reached convergence, i.e. only by tracing back from every state and comparing the results convergence can be detected. As a result, in order to avoid truncation noise the Trellis depth needs to be selected with an additional safety margin, such that the probability of reaching convergence is very large. This reduces the efficiency of the Viterbi approach further.

In an implementation that follows the generic pruned look-ahead approach, the output symbol can be found by reading out a single memory location, and convergence can be detected easily as well. This is accomplished by storing the full history, i.e. the last L feedback values, directly with each state instead of a pointer to the previous state as is done in

the Viterbi approach. At every time step a new value is added to the (circular) buffer. The last entry of *any* of the 2^N buffers can be directly selected as the output symbol. Furthermore, by simply comparing the last symbol of all the buffers convergence can be detected.

The total amount of memory required to store the history of each state directly is exactly equal to the amount of memory required for storing the backtracking information, i.e. in both cases L bits of memory are required per state, resulting in a total of $2^N \cdot L$ bits. The penalty of storing the complete history per state is that, depending on what solutions are selected to continue, the complete history of a state may need to be duplicated. More specifically, no copies of the history of a state are required as long as the state is only used once in the next time step. If a state is required twice, because it is appended with both a '1' and a '0' symbol, an additional copy of the history buffer is required. For every state that is used twice as source state there is another state that is not used, and the history buffer of this unused state can be used for storing the copy.

7.7 Conclusions

The Trellis sigma-delta modulation algorithm can be described in two ways. There is the original version of the algorithm as invented by Kato, in which the system is modeled as a HMM and where a Viterbi traceback is performed to realize look-ahead. Alternatively, the algorithm can be explained using the earlier introduced generic theory of pruned look-ahead modulation.

In the Trellis sigma-delta modulation algorithm, there are two parameters that determine how much look-ahead is realized. The number of parallel solutions that are maintained is equal to 2^N, where N is the Trellis order. The history length that is maintained before an output symbol is determined, or the latency of the algorithm, is specified by the Trellis depth L.

In order to advance time in the algorithm, for each of the 2^N parallel solutions the cost is calculated for appending the solution with a '+1' and a '-1' symbol. From the 2^{N+1} resulting potential solutions a selection is made, such that the 2^N remaining solutions cover all possibilities when only the most recent N bits are considered. The output symbol is found by determining what symbol was appended L time steps ago to the final solution. Depending on how the algorithm is implemented, this task requires tracing back L time steps using the Viterbi algorithm, or it

can be realized by simply reading a fixed location of a history buffer. The latter approach has several advantages, e.g. less computations and the possibility to detect if the algorithm has converged with minimal overhead.

Because the speed of convergence is not only a function of the loop-filter configuration and the Trellis order, but is also a function of the input signal, it is difficult to mathematically derive the required amount of history length. By means of simulations it has been determined that, for example, a history length of 1000 bits is sufficient to avoid truncation noise for typical fifth order loop filters with Trellis order $N = 10$. In general it holds that lower Trellis orders require less latency, and that more aggressive filters require a larger history length.

In ch. 5 a linear model, describing the NTF and STF of a generic look-ahead SDM, was derived. This model has been verified against practice, and was found to be fairly accurate. As predicted by the model, the SNR of a Trellis SDM is lower than that of a normal SDM if the same loop filter is used. The shape of the NTF is predicted with good accuracy, but the absolute levels in the baseband region can be off by several decibels, depending on the input signal level. The STF of a feed-forward Trellis SDM was found to be equal to unity, as predicted. However, the actual STF of a feed-back Trellis SDM deviates slightly from the predictions.

The classical functional performance of a Trellis SDM as a function of the Trellis order has been investigated by means of simulations. As expected, the SNR is not influenced by the Trellis order, but the suppression of distortion components scales with the Trellis order. Improvements in the THD of up to 20 dB can be realized by employing Trellis order $N = 10$. The SFDR can, typically, be improved by several dBs, until it is limited by the quantization noise-floor. For N as small as four the SINAD becomes equal to the SNR. There is no need to dither a Trellis SDM, since a low Trellis order will already result in a clean output spectrum without distortion.

As predicted in ch. 5 the stability of a Trellis SDM is a function of the Trellis order. Depending on the loop-filter configuration, an increase in the stable input range of 10 to 20% can realized by using a Trellis order $N = 10$. Unexpectedly, this increase in stable input range does not cause an increase of the peak SNR of the converter, which is typically realized at an input level that is independent of the Trellis order. However, the increase in stability does enable the use of more aggressive loop filters, which do cause an increase in the SNR of the converter. The SNR that can be realized in this fashion becomes, typically, equal to or higher than that of a normal SDM for the case $N = 1$, and up to 10 dB higher for a

large Trellis order.

By employing a high order Trellis SDM instead of a normal SDM, the problem of noise-modulation is reduced, but not solved, Whereas a normal SDM has a significant number of DC levels that cause idle tones, a Trellis SDM with $N = 10$ has very few. By using a Trellis SDM the variation in the output noise-level is also reduced, but still an increase of noise is present for high input levels compared to low input levels.

Overall, it is clear that the application of a high Trellis order brings significant advantages. Compared to a normal SDM a much higher signal conversion quality can be realized. However, for the Trellis sigma-delta modulation algorithm to be truly effective, a Trellis order of $N = 10$ or better is required. The computational load of such a converter is more than three orders of magnitude higher than that of a normal SDM, making the application of such a Trellis SDM not very practical. A further reduction of the number of parallel solutions is required, and ways to realize this are investigated in the next chapters.

Chapter 8

Efficient Trellis Sigma Delta Modulation

The Trellis sigma-delta modulation algorithm described in the previous chapter shows a clear improvement in performance compared to traditional sigma-delta modulation. However, this improvement comes at the cost of large computational load. As a result, practical implementations can only use a limited Trellis order, resulting in a performance that is far less than possible. In sec. 8.1 it will be shown that it should be possible to reduce the computational load of a Trellis SDM significantly, without causing a detectable reduction of the signal conversion quality. The resulting algorithm, called the Efficient Trellis sigma-delta modulation algorithm, is discussed in sec. 8.2. The algorithm has two main parameters, i.e. the Trellis order N and the number of parallel paths M, that determine the computational load and the performance of the converter. In sec. 8.3 an investigation is made on how many paths M there are required to realize the same performance as a full Trellis converter. The required history length, which is a function of both N and M, is explored in sec. 8.4. On the basis of these results, in combination with the insights from the previous chapter, the performance figures of an Efficient Trellis SDM are derived in sec. 8.5. The pruning of the solution space, which is key to a higher processing efficiency, introduces a sorting action in the algorithm. As a result, efficient sorting is required for an efficient implementation of the algorithm, and is discussed in sec. 8.6. Finally, conclusions are drawn in sec. 8.7.

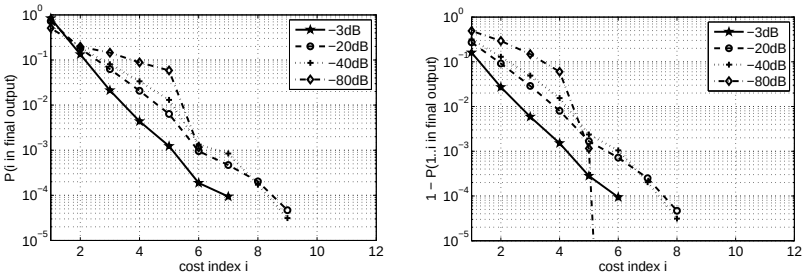
8.1 Reducing the number of parallel paths

In the original Trellis sigma-delta modulation algorithm at every time step there are 2^{N+1} cost values calculated, from which 2^N solutions are selected. However, a significant part of these calculations does not contribute to the final solution, since the feed-back patterns that are investigated do not match the input signal at all. For example, although the input signal is positive a feed-back sequence which represents a negative signal is evaluated, or a large amplitude feed-back sequence is evaluated while the input signal is small. Therefore, it should be possible to obtain a similar result as realized by the Trellis sigma-delta modulation algorithm while performing less computations.

An investigation is made on which and how many of the intermediate parallel solutions contribute to the final solution. This is done by creating a Trellis structure with $L = 64000$ in which the accumulated path cost for all the 2^{N+1} possibilities for continuing the Trellis are stored for each of the 64000 time steps. It is now possible to order, for every time step, the 2^{N+1} cost values and to calculate the cost index of each state, i.e. the state with the lowest accumulated path cost has index 1 and the state with highest cost has index 2^{N+1}. During the trace-back procedure the cost indices of the states that are passed are recorded. From this data the probability that a state will contribute to the final solution, based on its cost index, is calculated.

The experiment, as described above, is performed for a Trellis order of $N = 4$ for SDM configuration SDM2 (see app. B). The input signal consists of a 1 kHz sine wave at various amplitudes, ranging from -80 dB to -3 dB. The probability that a path is part of the final solution is plotted in fig. 8.1(a). For the -3 dB input signal the recovered output signal passes through the cheapest state in 84% of the time, whereas for the -80 dB signal the cheapest state is only selected 51%. The probability that the state with cost index six is selected is below $2{\cdot}10^{-3}$, independent of the input level. The most expensive state that was part of the output sequence has cost index nine. Thus, if it would be possible to only calculate the nine cheapest paths at every time step, the same solution would be found as if all the 32 (2^{4+1}) paths are calculated.

An an alternative to calculating the probability that a path is selected, it is also possible to calculate the probability that the first n paths are not part of the final solution. From this curve, depicted in fig. 8.1(b), it can be found how many paths are, on average, required to obtain a certain error probability, i.e. the probability that the original solution can not be constructed from the n paths.

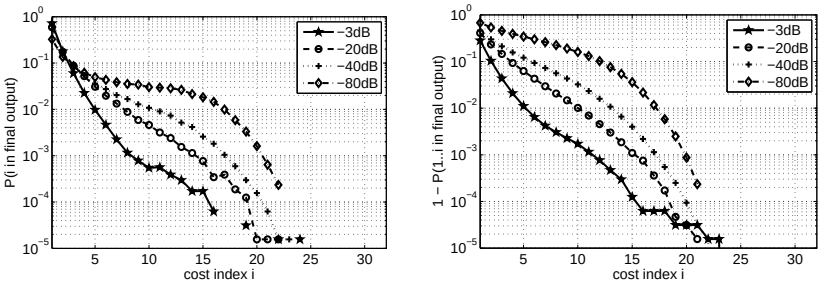


(a) P(i in final solution) (b) P(1..i not in final solution)

Figure 8.1: Probability that a path is selected for the final solution as a function of its cost index (a) and the probability that that the first n paths are not within the final solution (b) for SDM configuration SDM2 with N=4. The input signals are 1 kHz sine waves at various levels.

The same experiment is also performed for $N = 8$, with the results shown in fig. 8.2(a) and (b). Whereas in the $N = 4$ case the probability that a path is selected for the final output decreases very fast as the cost increases, in the $N = 8$ case the probability reduces only very slowly, especially for low input amplitudes. At least the 18 cheapest paths need to be evaluated in order to reach an error probability of 10^{-2}. From the 24 best paths the final solution is found, while 512 (2^{8+1}) cost values are calculated at each time step. If it would be possible to only calculate these 24 solutions a speed-up of 21 times would be realized, without any performance degradation.

The experiments described above confirm the hypothesis that, especially for high Trellis orders, many solutions are investigated that do not contribute to the final solution. Only the solutions that have a path cost that is low compared to the other potential solutions have a large probability of being selected for the output. Once a path accumulates a large cost the probability that it will be part of the output reduces severely, and there is no reason to continue with the path. However, when a path with a low cost is extended it can become expensive, but it can also maintain a low cost. Thus, when a path is extended with a symbol and the result is a path with a relatively low cost, the path had already a relatively low cost before the new symbol was added. If a path is expensive, it is either because it is originating from an expensive path, or because the newly added symbol forms a bad match with the input signal.



(a) P(i in final solution) (b) P(1..i not in final solution)

Figure 8.2: Probability that a path is selected for the final solution as a function of its cost index (a) and the probability that that the first n paths are not within the final solution (b) for SDM configuration SDM2 with N=8. The input signals are 1 kHz sine waves at various levels.

From the above it can be concluded that in order to maintain the n paths with the lowest cost, m parallel paths are required, with $m \geq n$. The actual number of paths that need to be maintained should be larger than the wanted number of cheapest paths, since not every path with a low cost value will result in a new path with a low cost. Once the number of parallel paths is large enough the original performance of the Trellis SDM will be achieved and no improvement in performance can be realized by increasing the number of paths. Thus, by adapting the Trellis algorithm to maintain a fixed number of paths M, with $M << 2^N$, virtually the same performance should be achievable as with the original algorithm.

8.2 Algorithm

An algorithm that investigates only a fraction of all the paths, as described in the previous section, has been realized and was coined Efficient Trellis sigma-delta modulation [20]. The algorithm is similar to the original Trellis algorithm, with the main difference that only M parallel paths are maintained instead of 2^N. Identical to the original algorithm, the N newest bits of all the parallel paths are forced to be different. This results in the following steps:

1. extend the M paths with a '0' and a '1' bit and calculate the $2M$ cost values;

2. calculate the $2M$ accumulated path cost values;

3. select the M unique paths with the lowest accumulated path metric and update the look-ahead filter states;
4. adjust the path metric values;
5. determine the output symbol;
6. invalidate the paths that have not converged.

In the first step of the algorithm, all the M parallel paths that are tracked are extended with the two possible symbols, and the cost for adding those bits is calculated. This step is in principle equal to the first step of the original Trellis algorithm, except that now only M paths are extended instead of 2^N paths.

Once the $2M$ cost values are calculated, in the second step of the algorithm the accumulated path cost for the $2M$ potential paths is calculated. This step is again similar to the operation that is performed in the original Trellis algorithm, with the difference that now only $2M$ accumulated path cost values are calculated instead of 2^{N+1}.

In the third step of the original algorithm 2^N times a choice is made between two paths, such that 2^N paths remain that are unique in their newest N symbols. In the efficient Trellis algorithm the operations performed for this step are different. More specifically, from the $2M$ potential solutions M solutions are selected, with the constraint that these M solutions have the lowest cost values, and that the M selected solutions are all unique in their newest N symbols. It is therefore not possible to select M times between two solutions. As a result, a two step approach is required. First, the $2M$ solutions are sorted based on their cost. In a second step the most expensive version of every solution that is present twice is removed. After this step at least M solutions and at maximum $2M$ solutions remain. The first M solutions of this list are the desired M unique paths with the lowest accumulated path metric, and it is now possible to update all the internal look-ahead filter states.

The fourth step of the algorithm is again identical to that of the original Trellis algorithm, and consists merely of subtracting the accumulated cost of the cheapest path from the cost of every path, such that the accumulated path metric stays bounded.

In the fifth step the output symbol is determined. Since the number of parallel paths M is limited and $M << 2^N$, backtracking using the Viterbi algorithm is highly inefficient. An implementation in which the complete history of a path is stored directly with the look-ahead filter is much more efficient, and, in addition, simplifies the last step of the

algorithm significantly. The output symbol is found by selecting the path with the lowest accumulated path score and selecting the symbol that was applied L clock cycles ago.

In the last step of the algorithm it is verified that all the paths have converged on the same output symbol, and otherwise corrective measures are applied. More specifically, if a path is not in agreement on the output symbol, the accumulated path metric of this path is increased by a large amount. As a result, in the next time step this path will not be part of the M cheapest paths, and the exploration of the path will be stopped. Although this test on convergence is not required if the history length L is large enough, it can be difficult to realize a sufficiently long history length efficiently. Especially in the case of a large Trellis order N, already with a small number of parallel paths M, the required history length becomes very large (sec. 8.4). As a result, with virtually no loss of performance, a higher computational efficiency can be realized by limiting the history length and testing for convergence.

8.3 Relation between N and M

From sec. 8.1 it is clear that only a fraction of all the 2^N paths that are maintained in the Trellis algorithm contribute to the final output solution. The Efficient Trellis algorithm attempts to exploit this fact and determines the output sequence on the basis of less parallel paths. However, the number of paths M that is required to achieve a similar performance with the Efficient Trellis algorithm as with the full Trellis algorithm cannot be determined from the experiments of sec. 8.1.

From ch. 7 it is known that the two biggest improvements realized by application of the Trellis algorithm are, typically, a reduction of distortion in the output signal and an improvement of the stability of the converter. Both these improvements become larger when a larger Trellis order is applied, and are, in principle, a suitable comparison criterion. However, not for all types of loop-filter configuration the improvement in THD or SFDR is present, and when it is present it is difficult to measure. Therefore, the comparison between the full Trellis algorithm and the Efficient Trellis algorithm will be based on the improvement in stability. The stability of a converter will be based on the maximum input amplitude that can be converted without causing instability to the converter.

For SDM configuration SDM2 (see app. B) the maximum input amplitude of a 1 kHz sine wave that can be converted without stability

problems is determined. The experiment is performed for several values of M for a Trellis order of 1, 2, 4, 8, 16, and 32. In fig. 8.3 the results are depicted for M equal to 1, 2, 4, 8, 16, and 128.

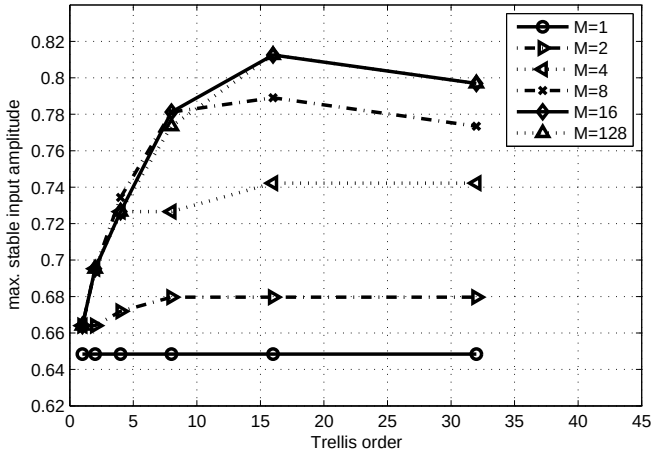


Figure 8.3: Maximum stable input amplitude as a function of the Trellis order for configuration SDM2 for a 1 kHz sine wave for various values of M.

For a constant value of M the stability of the converter, typically, becomes larger when the Trellis order is increased. For example, with $M = 4$ a maximum input amplitude of 0.74 can be handled if $N \geq 16$, but in the normal Trellis case with an equal number of parallel paths, i.e. $N = 2$, a maximum amplitude of only 0.695 can be converted. When $M = 16$ the maximum stability is realized for a Trellis order of $N = 16$, and the maximum level that can be converted is 0.815. An original Trellis with $N = 4$ maintains also 16 paths but only realizes a maximum input of 0.73. From fig. 7.21 it can be found that a full Trellis with $N = 10$ (1024 parallel paths) is stable up to an input level of 0.80. Thus, an Efficient Trellis SDM with $M = 16$ and $N = 16$ outperforms a full Trellis SDM with $N = 10$.

Although in general the stability of the converter increases when the Trellis order is increased and the number of paths is kept constant, a different behavior is present for $M \geq 8$ when N is increased from 16 to 32. Unexpectedly, the maximum input amplitude that can be handled reduces slightly for these cases. This is strange since for smaller values of M always an increase in stability is observed when N is increased.

A possible explanation for this phenomenon could be the following. In principle, a converter with $N = 32$ can generate the same solutions as a converter with $N = 16$, because the constraint imposed on the number of equal bits by $N = 16$ is stronger than the constraint imposed by $N = 32$. However, in the case of $N = 16$, sometimes a selection between two paths that are equal in their newest 16 bits will be made. As a result, the more expensive path that is rejected will be used to explore an alternative solution in the next clock cycle, instead of investigating a very similar solution as would happen in the case of $N = 32$. It is therefore imaginable that, with the limited number of parallel paths available, in the case of $N = 16$ more diverse solutions are investigated, which results in a larger stability. If the number of paths would be increased significantly, it is expected that a converter with $N = 24$ or $N = 32$ would realize a larger stability.

From fig. 8.3 it can also be seen that no performance improvement is realized when the number of parallel paths M is increased from 16 to 128, i.e. the two curves overlap each other for large N. Apparently, with $M = 16$ all the relevant solutions are found. From fig. 8.2 it is known that in a full Trellis with $N = 8$ the output sequence can pass through any of the 24 cheapest states, suggesting that at least 24 parallel paths are required to generate these solutions. However, the results shown in fig. 8.3 indicate that with 16 parallel paths a similar solution is found as with 128 parallel paths, and that this solution is of a higher quality than what is achieved with a full Trellis with $N = 8$. Thus, it can only be concluded that the more expensive paths that are occasionally selected in the full Trellis algorithm do not significantly contribute to a higher quality of the solution, and only a small number of paths is sufficient to realize an improvement in performance.

On the basis of the results shown in fig. 8.3 it is clear that the maximum stability for a given Trellis order is already reached when far less than 2^N paths are used. More specifically, for the case $N = 4$ the maximum stability is reached for $M = 4$, for $N = 8$ the maximum is obtained with $M = 8$, and for $N \geq 16$ it is sufficient to use $M = 16$ paths.

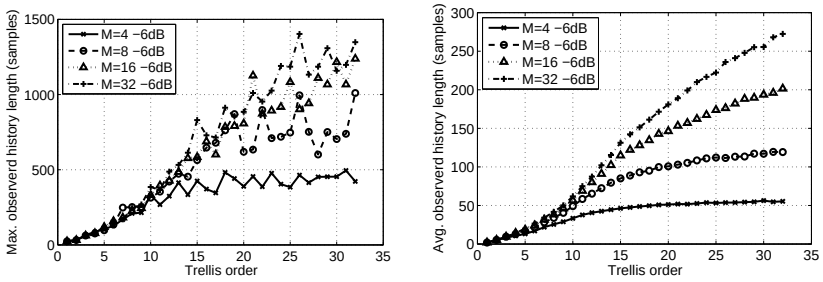
8.4 Required history length

Although an Efficient Trellis SDM is similar to a normal Trellis SDM, it is expected that the history length that is required for such a converter is slightly different because fewer parallel paths are present. Therefore, by means of simulations the minimum required history length of an Efficient Trellis SDM is investigated. The procedure followed is similar to the one

outlined in sec. 7.4 for a normal Trellis SDM.

From sec. 7.4 it is known that for a normal Trellis SDM the Trellis order and the signal amplitude have a big influence on the required history length, while the loop-filter configuration has only a minor influence and that the signal frequency has no influence on the required history length. In the case of an Efficient Trellis SDM a similar behavior is expected, except that there is also the number of paths M that will influence the required history length. For loop-filter configuration SDM2 (see app. B) the required history length is determined.

For a -6 dB input signal the history length is measured for Trellis orders of 1 up to 32 for M equal to 4, 8, 16, and 32. The maximum and average observed history length are depicted in fig. 8.4. For N up to 10 the maximum observed history length is equal for all values of M, and shows a behavior very similar to that of a full Trellis (see fig. 7.13). For larger values of N a clear difference between the different values of M can be observed, i.e. the larger the number of parallel paths the larger the required history length. For the case of $N = 32$ with $M = 32$ a maximum history length of approximately 1400 bits is found, while the average history length is 275 bits.



(a) Maximum required history length. (b) Average required history length.

Figure 8.4: The maximum (a) and the average (b) observed required history length for loop-filter configuration SDM2 as a function of the Trellis order for various values of M for an input amplitude of -6 dB.

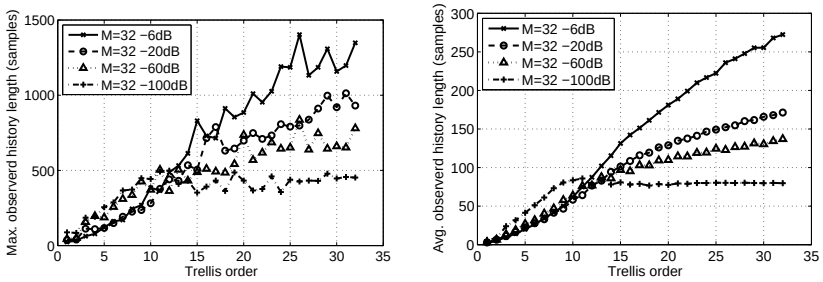
The same experiment as described above is again performed, but now the number of paths M is kept constant at 32 and the input amplitude is varied. The results are shown in fig. 8.5. The behavior for $N \leq 10$ is similar to that of a full Trellis SDM (fig. 7.13), i.e. the smaller the amplitude the longer the required history length. For larger N, i.e. $N > 12$, the situation is different, and a larger history length is required

for the large amplitude signals than for the low amplitude signals.

From the results discussed above it can be concluded that the required history length increases both with the number of parallel paths and with the Trellis order. However, depending on the Trellis order, it is either for small or for large signals that the maximum history length for unambiguously determining the output symbol is required. In the case of low Trellis orders the largest history length is required for small signals, but the difference in the required history length between small and large signals is relatively small. In the case the Trellis order is large, i.e. larger than 12, large amplitude signals require the largest history length, and the difference in the required history length between a large and a small signal can become as large as a factor four.

This difference in behavior can be explained as follows. A normal 1-bit SDM, typically, generates significant harmonic distortion for large amplitude signals. If an Efficient Trellis SDM is excited with a large amplitude signal, it will explore several potential solutions and search for the solution with the least amount of distortion. It is reasonable to assume that several solutions exist that encode the input signal with a reasonable quality. However, only over a relative long time span it will be possible to detect which sequence results in the least amount of distortion. If N is small, it will be difficult to realize such sequences, and as a result a relatively short history length is observed. If N is large, there is a large freedom in the generation of bit sequences, and similar solutions can be explored over many clock cycles. In this case it holds that the larger the signal amplitude is, the more difficult it is to find the best match, and the longer the required history length becomes.

The exception to the reasoning above seems to be the combination of a small amplitude signal with a small value of N, since now a larger history length is required than for a large amplitude signal in combination with a small value of N. In this case another effect is dominant. When the signal amplitude is small, many very different solutions exist that describe the signal with a good quality, which results in a large required history length. If the Trellis order is increased, more similar solutions are allowed, but since the number of parallel paths is limited, these solutions can not be explored and the observed history length does not increase further. Only if more parallel paths would be used the observed history length would increase further, until the next limit would be reached.



(a) Maximum required history length. (b) Average required history length.

Figure 8.5: The maximum (a) and the average (b) observed required history length for loop-filter configuration SDM2 as a function of the Trellis order for various input levels with $M = 32$.

8.5 Functional performance

The functional performance of an Efficient Trellis SDM is investigated on the basis of the classical signal quality indicators SNR, SINAD, THD, and SFDR. Next to these indicators, the SDM specific performance measures stability and noise modulation are investigated. Finally, a summary is given of the Efficient Trellis SDM functional performance.

8.5.1 SNR, SINAD, THD and SFDR

In sec. 7.5.1 the SNR, SINAD, THD, and SFDR performance of a variety of SDM configurations has been studied for a normal Trellis SDM. It was found that there is no significant difference between feed-forward configurations and feed-back configurations. Furthermore, it was shown that loop filters without resonator sections react slightly different to an increase in the Trellis order than loop filters with resonator sections. The order and aggressiveness of the loop filter was not of influence, i.e. the trend of improvement was the same for all loop-filter configurations with resonator sections and the same for all loop-filter configurations without resonator sections. Since an Efficient Trellis SDM is a derivative of a normal Trellis SDM the same type of behavior is found for an Efficient Trellis SDM. As a result it is only required to study the detailed behavior of an SDM without resonator sections, i.e. SDM configuration SDM1, and an SDM with a loop filter with resonators (SDM configuration SDM2, see app. B).

SDM1

In the case of a normal Trellis SDM the SNR, SINAD, and SFDR performance is virtually unaltered when the Trellis order is increased from $N = 1$ to $N = 8$. This is as expected since the in-band noise at the end of the pass-band is dominating (see fig. 8.6 for an example spectrum), and the same behavior is present for an Efficient Trellis SDM. The THD of the Trellis SDM is varying between 140 dB and 160 dB as a function of the Trellis order, but no improvement can be realized compared to $N = 1$. In the case of an Efficient Trellis SDM the situation is different, and it is possible to realize an improvement in the THD for a fraction of the computational cost.

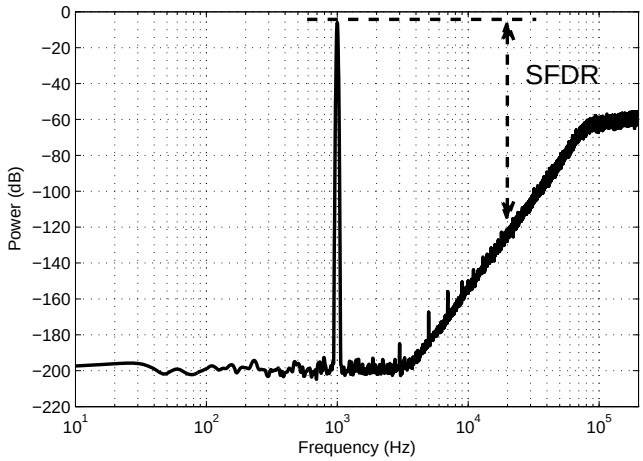
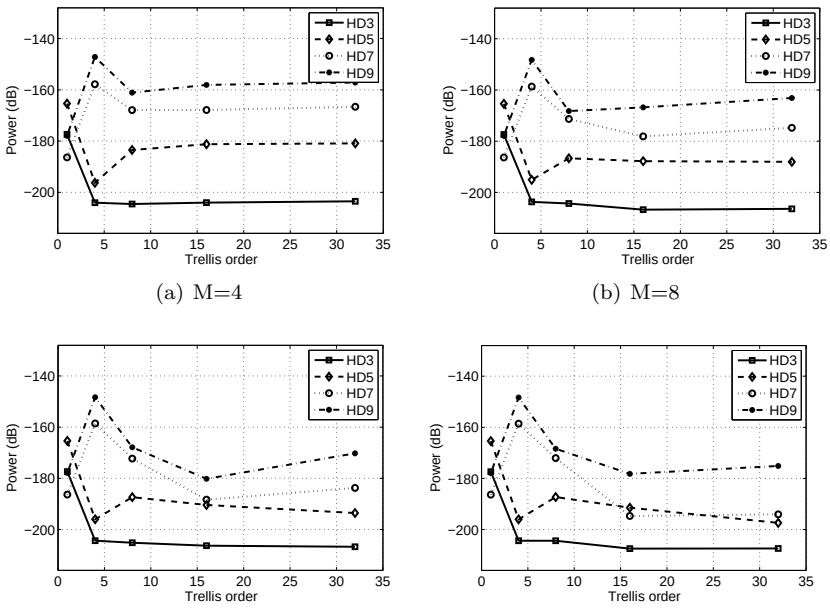


Figure 8.6: Example output spectrum of a fifth order SDM without resonator sections. The SFDR is limited by the quantization noise at the end of the pass-band. The harmonics are at much lower lower level, resulting in a much lower THD value.

In fig. 8.7 the power in the first four odd harmonics is plotted as a function of the Trellis order N for M equal to 4, 8, 16, and 32 paths ($L = 4096$). In the case of $M = 4$ and $M = 8$ no improvement in the THD is realized, since the biggest harmonic component does not reduce to below the value obtained for $N = 1$, i.e. for $N > 1$ the value of HD9 is the dominant harmonic, and is at a level above the value of HD5 for $N = 1$. However, in the case of $M = 16$ and $M = 32$ the power in the



(a) M=4 (b) M=8

(c) M=16 (d) M=32

Figure 8.7: Power in HD3, HD5, HD7, and HD9 for $M = 4$ (a), $M = 8$ (b), $M = 16$ (c), and $M = 32$ (d) for configuration SDM1 as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB.

harmonics reduces for all $N \geq 8$. A minimum is realized for $N = 16$, while the result for $N = 32$ is of a minimally lower quality. The resulting SFDR, in combination with the SNR, SINAD, and THD, are depicted in fig. 8.8 for $M = 32$. The reduction of the harmonics results in a total improvement of the THD of just over 10 dB compared to the setup with $N = 1$, at the cost of only 32 parallel paths.

SDM2

For a typical SDM with resonator sections the harmonic distortion, most notably the third order harmonic, is limiting the SFDR significantly (see fig. 8.9). In the case of loop-filter configuration SDM2 application of the original Trellis sigma-delta modulation algorithm can improve the SFDR by 20 dB and the THD by 23 dB, by using 64 parallel paths ($N = 8$ in fig. 7.18).

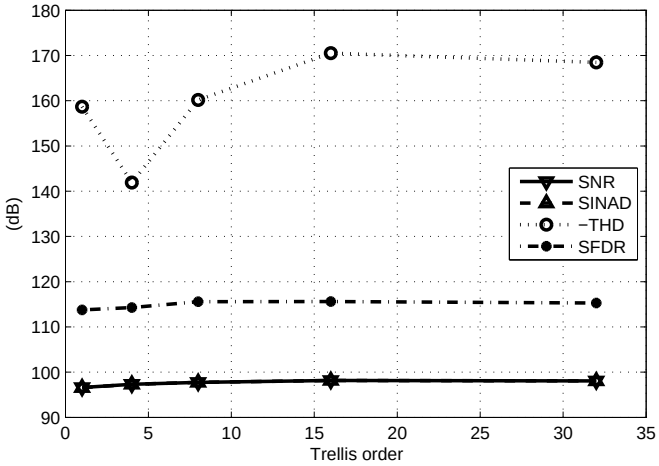


Figure 8.8: SNR, SINAD, THD, and SFDR performance for configuration SDM1 as a function of the Trellis depth with $M = 32$ for a 1 kHz sine wave with a power of -6 dB.

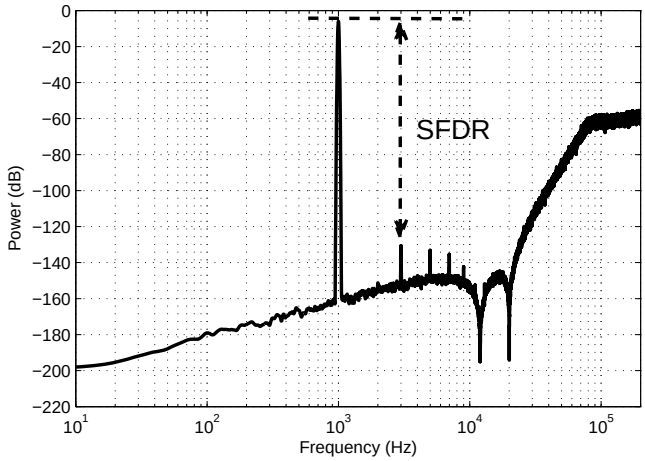
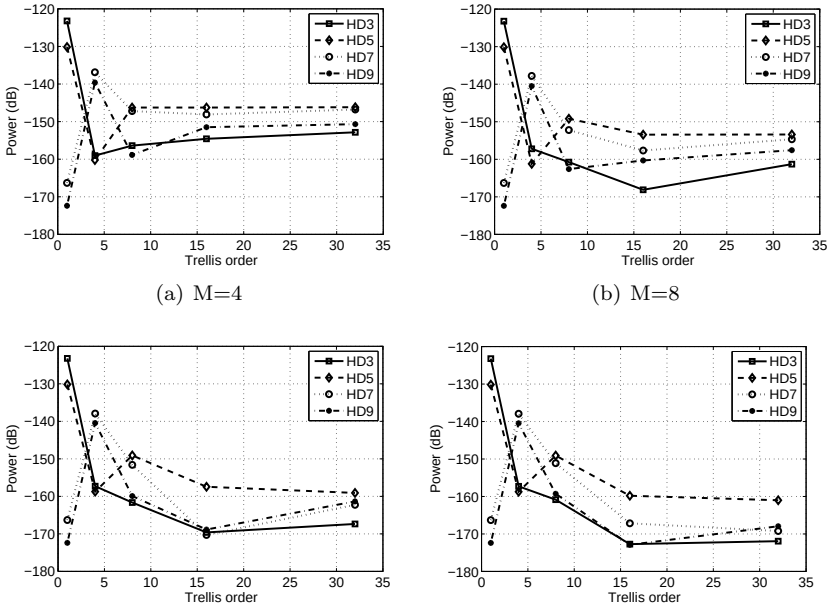


Figure 8.9: Example output spectrum of a fifth order SDM with two resonator sections. The SFDR is limited by the third harmonic. The THD is slightly lower than the SFDR, since also higher order harmonics are present.



(a) M=4

(b) M=8

(c) M=16

(d) M=32

Figure 8.10: Power in HD3, HD5, HD7, and HD9 for $M = 4$ (a), $M = 8$ (b), $M = 16$ (c), and $M = 32$ (d) for configuration SDM2 as a function of the Trellis depth for a 1 kHz sine wave with a power of -6 dB.

At a fraction of the cost of a full Trellis SDM an Efficient Trellis SDM can realize a bigger improvement, as shown in fig. 8.10 as a function of the Trellis order N ($L = 4096$). For example, with 8 parallel paths ($M = 8$, fig. (b)) all the harmonics are suppressed to a level below -150 dB for $N \geq 16$, which is better than what is achieved by the full Trellis with $N = 8$ that needs $2^8 = 256$ parallel paths. Independent of the number of parallel paths, a higher Trellis order results in a larger suppression of the harmonics. As a result, the total power in the harmonics is decreasing when N is increased, and the distribution between the power in the harmonics is equalized.

In fig. 8.11 the resulting SNR, SINAD, THD, and SFDR curves are shown for $M = 32$ as a function of the Trellis order. The improvement in SFDR is equal to that obtained by the full Trellis, since this is limited by the quantization noise floor once the harmonics are suppressed enough. The THD is improved by 37 dB to a level of -153 dB for $N = 32$.

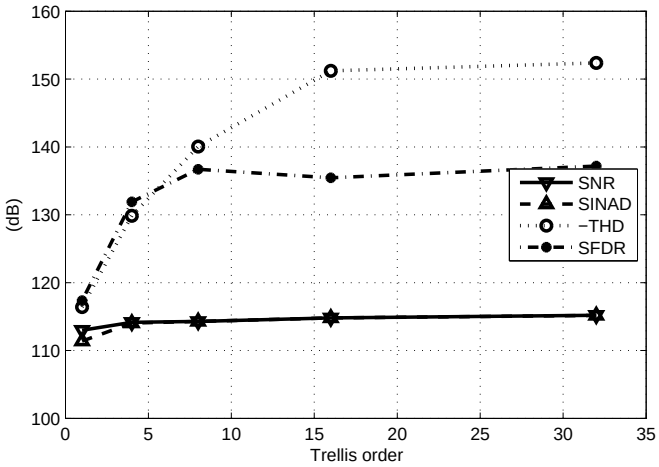
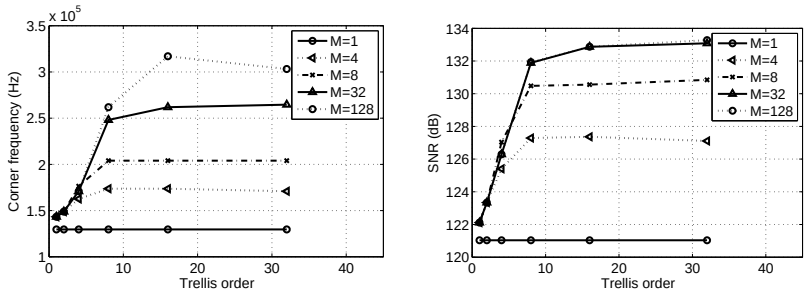


Figure 8.11: SNR, SINAD, THD, and SFDR performance for configuration SDM2 as a function of the Trellis depth with $M = 32$ for a 1 kHz sine wave with a power of -6 dB.

8.5.2 Converter stability

The stability of an Efficient Trellis SDM, expressed as the maximum input amplitude that can be converted without causing instability to the converter, has already been investigated and compared to that of a normal Trellis SDM in sec. 8.3. The alternative stability measure, i.e. the maximum loop-filter corner frequency that can be used with a given input signal, will be investigated in this section.

For loop-filter configuration SDM2 (see app. B) the maximum loop-filter corner frequency is measured for a -6 dB 1 kHz sine wave. The experiments is performed for several combinations of the Trellis order N and the number of paths M. The maximum corner frequency, as well as the SNR that is achieved for the maximum corner frequency, is depicted in fig. 8.12. In general, a higher corner frequency is achieved for larger values of M, as long as N is large enough. For $M \leq 8$ the maximum corner frequency is achieved for $N \geq 8$. For the case of $M = 32$ a maximum corner frequency of approximately 260 kHz is achieved for $N = 32$, and nearly the same for $N = 16$. For $M = 128$ it is found that the maximum corner frequency reduces from 320 kHz to 300 kHz when N is increased from 16 to 32. A similar reduction in stability was



(a) Stable corner frequency.

(b) SNR.

Figure 8.12: Maximum loop-filter corner frequency that results in stable operation as a function of the Trellis order for various number of paths for a -6 dB 1 kHz sine wave with filter configuration SDM2 (a) and the associated SNR (b).

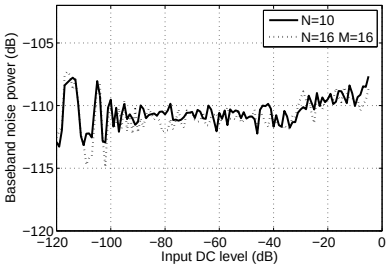
also observed in the experiments described in sec. 8.3. However, for the resulting SNR this has no impact, i.e. the SNR is equal for $N = 16$ and $N = 32$. A close inspection of the results reveals that also the achieved SNR for $M = 32$ and $M = 128$ is equal for all values of N. Thus, independent of the actual corner frequency, the SNR does not improve for cutoff frequencies above approximately 300 kHz. Such a saturation of the SNR was also recorded for the normal Trellis SDM (sec. 7.5.2). In chapter 12 this phenomenon is studied in more detail.

A comparison of the maximum achieved corner frequency with that of a normal Trellis SDM reveals that the maximum achieved value of the Efficient Trellis SDM is lower than that of the normal Trellis SDM, i.e. the Trellis SDM is stable with a corner frequency of 420 kHz for $N = 10$ whereas the Efficient Trellis SDM only reaches a maximum corner frequency of 320 kHz. However, in the case of the normal Trellis SDM the maximum SNR that is achieved is 132 dB, while the Efficient Trellis SDM realizes the slightly higher SNR of 133 dB with a lower corner frequency, at a fraction of the computational cost.

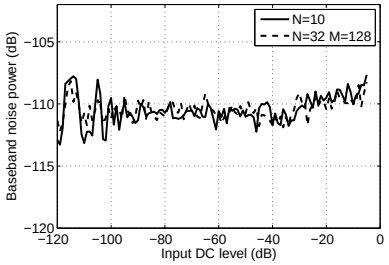
8.5.3 Noise modulation

The in-band noise modulation of an Efficient Trellis SDM is investigated by measuring the amount of noise present in the 0-20 kHz band for various DC input levels. The experiment is performed for loop-filter configuration SDM1 (see app. B) for an Efficient Trellis SDM configured

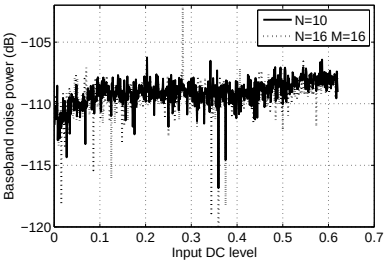
with $N = 16$ and $M = 16$, as well as for a configuration of $N = 32$ and $M = 128$. The amount of noise modulation is compared to that of a normal Trellis SDM with $N = 10$. Two experiments are performed, i.e. one with a logarithmic selection of the DC levels and one with a linear selection. The logarithmic selection is useful to illustrate how the amount of in-band quantization noise varies globally as a function of the amplitude of the input signal. The linear amplitude selection criterion results in rational DC levels, which will often trigger limit cycles in a typical SDM.



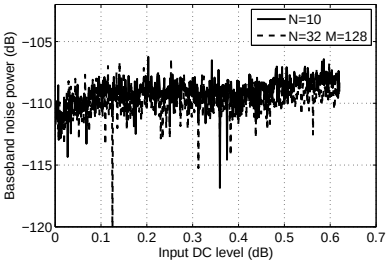
(a) $N = 10$ vs. $N = 16$ $M = 16$ (log sweep)



(b) $N = 10$ vs. $N = 32$ $M = 128$ (log sweep)



(c) $N = 10$ vs. $N = 16$ $M = 16$ (linear sweep)



(d) $N = 10$ vs. $N = 32$ $M = 128$ (linear sweep)

Figure 8.13: In-band noise as a function of the DC input level. Comparison of a Trellis SDM with $N = 10$ to an Efficient Trellis SDM with $N = 16$ and $M = 16$ in fig. (a) for a logarithmic input level selection and in fig. (c) for a linear input level selection. Same comparison to an Efficient Trellis SDM with $N = 32$ and $M = 128$ in fig. (b) and fig. (d) for a logarithmic, respectively, linear input level selection.

In the first experiment the DC levels are selected in the range from -120 dB to -5 dB in steps of 1 dB. The results for the Trellis SDM with

$N = 10$ and the Efficient Trellis SDM with $N = 16$ and $M = 16$ are plotted in fig. 8.13(a). The in-band noise level for both converters is nearly the same, with the Trellis SDM showing minimally less variation in the in-band noise. In fig. 8.13(b) the Trellis SDM result is compared to that of an Efficient Trellis SDM with $N = 32$ and $M = 128$. With this configuration of the Efficient Trellis SDM the two modulators realize the same amount of in-band noise.

In the second experiment the DC levels are selected as multiples of $\frac{1}{1024}$. The comparison of the Trellis SDM and the Efficient Trellis SDM with $N = 16$ and $M = 16$, depicted in fig. 8.13(c), confirms that the output of the Efficient Trellis SDM has slightly more variation in the noise level than the Trellis SDM. In the figure this can be recognized as several spikes, both up and down, in the in-band noise-level of the Trellis SDM. When the Efficient Trellis SDM is configured with $N = 32$ and $M = 128$, as shown in fig. 8.13(d), nearly all the spikes disappear. However, for a DC level of $\frac{1}{8}$ a new downward spike has appeared, and for this specific DC level the noise-level reduces to -124 dB.

If the noise-level increases, i.e. upward spikes in the plots, the SNR reduces. If the noise-level decreases, a smaller conversion error is realized and a higher SNR is resulting, and thus, in principle, a better conversion quality is realized. An (Efficient) Trellis SDM attempts to realize the best match between the input signal and the encoded representation, such that the resulting SNR is as high as possible. With more parallel paths more potential solutions can be investigated and, typically, a higher conversion quality is realized. With this insight it can be understood that the converter with $M = 128$ realizes a baseband noise-floor that is, on average, lower than the converter with $M = 16$. For the input level of $\frac{1}{8}$ a special solution is found, i.e. a limit cycle is realized that results in a DC level of $\frac{1}{8}$ and that has hardly any low-frequency content. Although this solution does not result in a typical noise-shaped spectrum, it is a solution that results in a higher SNR than what would be resulting from a normal noise-shaped spectrum. Thus, in principle, a solution with a higher than average conversion quality is realized. However, in terms of noise-modulation performance a lower quality is achieved, since here the objective is to have a constant noise-floor. A consequence of this is that an Efficient Trellis SDM, because it is able to realize a higher conversion quality for typical signals than a traditional Trellis SDM, will exhibit stronger noise-modulation for rational DC levels. Thus, although on the basis of the first experiment it can be concluded that the global noise modulation behavior of the Trellis SDM and the Efficient Trellis SDM are equal, the overall noise modulation properties of the Efficient Trellis SDM are, as expected, of

a slightly lower quality because of the better signal encoding quality for rational DC levels.

8.5.4 Summary

The SNR, SINAD, THD, and SFDR performance of an Efficient Trellis SDM has been investigated as a function of the Trellis order N and the number of Trellis paths M. In the case of loop-filter configuration without resonator sections the SNR, SINAD, and SFDR are insensitive to the value of N and M, equal to the situation with a normal Trellis SDM. However, with an Efficient Trellis SDM it is possible to improve the THD while in the case of a normal Trellis SDM no improvement can be realized for practical values of N. More specifically, with $M \geq 16$ a decrease in the THD is realized for $N \geq 8$, compared to the situation of $N = 1$. For $N = 16$ the biggest improvement in THD is realized, with only a small reduction for $N = 32$. With $N = 16$ and 32 parallel paths the THD is improved from -160 dB to -170 dB. The situation for a loop filter with resonator sections is different, i.e. an improvement in the SFDR and THD can in this case be realized already with very few parallel paths. For example, with $M = 4$ and $N \geq 8$ the power in the harmonics is suppressed by more than 20 dB compared to the situation of $N = 1$. If 32 parallel paths are used the THD is improved by approximately 35 dB to a level of -152 dB. Independent of the number of paths employed, the suppression of distortion improves when a larger value of N is used. The difference in performance between $M = 16$ and $M = 32$ is very small, and in both cases a higher level of performance is achieved than with a normal Trellis SDM with $N = 10$.

The stability of an Efficient Trellis SDM, expressed as the maximum signal amplitude that can be converted without causing instability, was already investigated in sec. 8.3 in order to determine how many parallel paths are sufficient. It was found that for loop-filter configuration SDM2, a fifth order loop filter with resonator sections, the maximum stability was realized for $N = 16$, and that there was no difference in performance between $M = 16$ and $M = 128$. With $M \geq 16$ the maximum signal amplitude was improved from 0.65 to 0.81 for $N = 16$. In the case of $N = 32$ a small reduction in stability is occurring and the maximum is reduced to 0.80. Compared to a normal Trellis SDM with $N = 10$ the stability of an Efficient Trellis SDM with $M = 16$ and $N = 16$ is better.

The alternative stability measure, i.e. the maximum corner frequency of the loop filter that can be used for a given input signal, has been investigated in this section, also for loop-filter configuration SDM2. Again,

for $N = 16$ the maximum stability is realized. However, in this case the stability does improve if more than 16 parallel paths are used. More specifically, a significant difference in the maximum corner frequency can be detected between $M = 16$, $M = 32$, $M = 64$, and $M = 128$. The Efficient Trellis SDM with $M = 128$ realizes a maximum corner frequency of 320 kHz with $N = 16$, whereas a normal Trellis SDM with $N = 10$ is stable for a corner frequency of 420 kHz. Thus, compared to a normal Trellis SDM the stability is less. However, when the SNR that is realized at the maximum stable corner frequency is compared, it is found that the Efficient Trellis SDM realizes an SNR (133 dB) that is slightly higher than that of the normal Trellis SDM (132 dB). Furthermore, no significant difference in the SNR can be detected for the case of $M = 32$ and $M = 128$. In both cases the maximum corner frequency is high enough to cause a saturation of the SNR.

The amount of variation in the baseband quantization noise of an Efficient Trellis SDM has been compared to that of a normal Trellis SDM. The global variation in the noise power, i.e. the difference in the noise level for very low power signals and large power signals, is very similar for a Trellis SDM with $N = 10$ and an Efficient Trellis SDM with $M \geq 16$. A comparison of the noise power curve of an Efficient Trellis SDM with $N = 16$ and $M = 16$ to that of an Efficient Trellis SDM with $N = 32$ and $M = 128$ reveals that the modulator with 128 parallel paths has slightly less variation in the noise power. This is most clearly visible when rational DC levels are supplied to the modulators, manifesting itself as a reduced number of peaks in the noise floor. However, the Efficient Trellis SDM with $M = 128$ has at least one DC level that results in a strong decrease of the in-band noise level, because a solution that is based on a limit cycle with little baseband content is found. Such a solution results in a higher SNR, and is therefore preferred by the modulator over a solution with a typical noise-shaped spectrum. In general, it can be expected that if a larger number of parallel paths is used, the Efficient Trellis SDM will realize a better conversion quality and more solutions for rational DC levels will be found that are based on a limit cycle. Thus, although the global variation in the baseband quantization noise is of an equal level for the Trellis SDM and the Efficient Trellis SDM, the Efficient Trellis SDM has a minor disadvantage over the full Trellis SDM when also rational DC levels are considered, since it is able to realize a higher SNR for this type of signals that results in a larger amount of variation. In practice, this disadvantage is hardly relevant since typical input signals are non-DC, and for other signals the amount of variation in the baseband quantization noise is on an equal level.

8.6 Implementation aspects

To realize an efficient implementation of an Efficient Trellis SDM the same procedure should be followed as for a normal Trellis SDM, i.e. it is important to have an efficient implementation of the look-ahead loop filter such that no unnecessary calculations are performed, and the output symbol selection should be realized by storing the history of each path directly with its look-ahead filter and reading out a memory location directly in order to avoid a trace-back action. However, unlike in the normal Trellis sigma-delta modulation algorithm, in the Efficient Trellis sigma-delta modulation algorithm the best M unique paths are selected from the $2M$ potential paths at every clock cycle, and a sorting and comparison operation is required. This step of the algorithm can easily become the most time consuming operation, especially for large values of N in combination with many parallel paths, and a good implementation efficiency can only be realized if the selection step is implemented properly.

8.6.1 Selection step

A straightforward implementation of the selection step can be realized by first sorting the $2M$ cost values, then removing the solutions that are not unique, and finally selecting the remaining best M solutions. However, such an approach will not result in an computationally efficient solution since a lot of unnecessary computations are performed.

In the straightforward approach, the number of computations required to sort all the solutions on their cost is $\mathcal{O}(2M \log 2M)$ at best. Examples of algorithms that reach this efficiency are the quicksort algorithm, heapsort algorithm, and the binary tree sort algorithm [66]. However, such an efficiency can only be realized when very large sets of data are sorted, because actual implementations of these algorithms have significant overhead. Since M is reasonably small, i.e. from the experiments it is clear that there is little benefit in having more than 32 parallel paths, the sorting algorithms that reach such a high efficiency, will not perform well.

Another disadvantage of most of the sorting algorithms that are efficient for large sets of data, is that with these algorithms it is not possible to obtain a partially sorted list, i.e. a list in which only the best n entries are determined and where the remaining entries are not yet sorted. Furthermore, none of these algorithms is able to add a single entry to an already sorted list without sorting the complete list again. As a res-

ult, for the Efficient Trellis sigma-delta modulation algorithm all the $2M$ cost values should first be calculated, then they can be sorted, and finally the non-unique solutions can be removed from the sorted list.

An efficient solution to the sorting problem can be realized by using a modified insertion sort algorithm. The normal insertion sort algorithm [40] has an average number of computations that scales $\mathcal{O}(n^2)$. However, for the Efficient Trellis sigma-delta modulation algorithm it is in most cases not required to sort all the $2M$ cost values completely. More specifically, if the list of sorted values contains M entries, a new entry does not need to be added to the sorted list if it has a higher cost value than the last entry in the sorted list. If the cost of the potential new entry is lower than the most expensive entry in the list, a second check is required to see if the potential new entry should be really added to the list. If the entry is unique, i.e. there is no other entry with the same newest N bits, the potential entry is insertion sorted. If the entry is not unique and the cost of the other entry with the same newest N bits is lower, the potential entry is not added. If the entry is not unique but its cost is lower than that of the entry with the same bitstream, the potential entry is insertion sorted, such that the more expensive entry with the same bitstream is removed from the list and the number of entries in the sorted list stays constant.

This approach results in a minimal amount of sorting, especially if the entries are added in a (near) sorted order. From sec. 8.1 it is clear that a path with a low cost, typically, results in two new paths with a low cost value, and that only occasionally a cheap path becomes expensive. Thus, if the sorted list is built, starting with the paths that are originating from the cheapest path and ending with the paths that are originating from the most expensive path, it is likely that few insertions are required to construct a sorted list. This can be realized at zero overhead cost, by maintaining the paths that proceed to the next time step in the order of the sorted list, i.e. the cheapest path will be processed first in the next time step and the most expensive path will be processed last. As a result of this approach, the computational complexity of the sorting becomes, on average, almost linear with the number of paths, i.e. $\Omega(2M)$, which is a near optimal result for non-real-time operation. If real-time operation is required, the worst-case performance of the sorting algorithm should be considered, and possibly better alternatives exist.

Although the average computational complexity of the sorting problem can be reduced to almost linear with the number of paths, the complexity

of the total selection problem stays $\Omega(M^2)$ for practical realizations[1], because of the check for uniqueness of the solutions that are added to the sorted list. Consider the case that the sorted list contains only one entry and that a second entry with a higher cost is added at the end of the list. Before the new entry can be added to the sorted list its bitstream should be compared to the bitstream of all the entries with a lower cost. For the second entry that is added this means it has to be compared to a single solution, the third entry to two solutions, and so on. Only in the case a matching bitstream is found the comparison loop can be aborted, since the new entry should not be added to the list. If the new entry is not added to the end of the list but at another position, it still needs to be compared to all the other solutions, i.e. to the cheaper solutions to see if the new entry can be added, and to the more expensive solutions to see if there is a more expensive duplicate that needs to be removed. As a result, in order to find the M unique solutions with the lowest cost, at least $1+2+..M-1 = \frac{M-1}{2} \cdot M = \frac{M^2-M}{2}$ bitstream comparisons need to be performed. For each cheaper path that is found after the first M entries are added to the sorted list, $M-1$ bitstream comparisons need to be performed, resulting in a maximum of $(M^2 - M) + M * (M - 1) = 2M^2 - 2M$ bitstream comparisons.

In the alternative approach of processing the solution that has the highest cost first instead of last, ideally, all the new entries will be added at the beginning of the sorted list, and it might seem that no bitstream comparisons are required. However, this is not the case because a comparison has to be made with all the more expensive solutions to determine if any of those should be removed because it is a more expensive duplicate. As a result, the number of bitstream comparisons is equal to before, but the cost for sorting the solutions increases since entries need to be added at the beginning of the sorted list. Thus, the best approach is to add entries in an ascending cost order.

8.7 Conclusions

In the original full Trellis algorithm only a fraction of all the calculated solutions is used to determine the actual final output symbol sequence. More specifically, only the solutions that have at any moment in time a path cost that is low, compared to other paths, have a large probability of becoming part of the output sequence. By modifying the Trellis

[1] it is possible to reduce the complexity of the test for uniqueness to $\mathcal{O}(2M \log 2M)$ by sorting the bitstreams as well, but the overall resulting complexity and overhead will be very high, making the approach not practical

algorithm to evaluate only the solutions with a low cost, i.e. the paths that have a large probability of belonging to the output sequence, a large reduction of the computational load can be realized, while maintaining virtually the same conversion quality as when evaluating all possible paths.

The resulting Efficient Trellis sigma-delta modulation algorithm has three parameters that control the computation load and conversion quality. The Trellis order N determines how many of the newest bits of all the solutions that are under investigation should be considered when testing for uniqueness of the solutions, the number of parallel solutions is specified by M, and with L the latency of the algorithm, required to realize convergence on the output symbol, is set. The computations that are required at each clock cycle to advance time and determine an output symbol are nearly the same as for the full Trellis algorithm, with the exception that instead of operating on 2^N parallel solutions there are M solutions to perform calculations on.

In the original Trellis algorithm the conversion quality, typically, improves when N is selected larger. In the Efficient Trellis sigma-delta modulation algorithm this also holds, but only if the number of parallel solutions M is large enough. However, for every value of N there is also a certain number of paths $M \ll 2^N$ that will realize the maximum performance for the specific value of N. In practice this means that for $N = 16$ the maximum performance is realized for a value of M between 16 and 32, and increasing the number of parallel paths further does not bring any significant improvement. In the original Trellis algorithm with $N = 16$ a total number of $2^{16} = 65\,536$ parallel paths would be investigated in order to realize the same performance.

Compared to the original Trellis algorithm the signal conversion performance of the Efficient Trellis sigma-delta modulation algorithm is, typically, equal or higher at a fraction of the computational load. For example, by using 16 parallel paths it is possible to improve the THD of a converter with a loop filter without resonator sections by 10 dB, while with a normal Trellis SDM the THD can not be improved when using as many as 1024 parallel paths. In the case of a fifth order loop filter with resonator sections the THD can be improved by more than 30 dB by selecting $N = 32$ with $M = 32$. The amount of in-band noise modulation of an Efficient Trellis SDM can be reduced to a similar level as that of a full Trellis modulator with $N = 10$ by using 64 to 128 paths.

Although far less parallel paths need to be maintained by the Efficient Trellis sigma-delta modulation algorithm than by the full Trellis algorithm, the computations of the Efficient Trellis sigma-delta modula-

tion algorithm are more complex. Whereas in the original algorithm at each time step 2^N times a selection between two potential solutions had to be made, in the efficient Trellis algorithm the M solutions with the lowest cost need to be selected. As a result of this, a sorting operation on the path cost values is required. Because the number of entries to sort is relatively small the traditional fast sorting algorithms that, typically, have significant overhead do not perform well. A modified insertion sort algorithm has been introduced that can achieve a computational complexity that is almost linear with the number of paths if all the solutions are processed in sorted order. However, the test for uniqueness of the selected solutions is of a complexity $\Omega(M^2)$. As a result, already for small values of M the computational load of the Efficient Trellis sigma-delta modulation algorithm is mainly determined by this test, especially if N is large. However, from a signal conversion perspective N is preferably large, resulting in a large computational load. Thus, in order to realize a look-ahead modulator with a higher computational efficiency and good signal conversion quality, it is desirable to change the test for uniqueness of the parallel solutions. Such a solution can be realized, as will be demonstrated in the next chapter.

Chapter 9

Pruned Tree sigma-delta modulation

From the previous chapter it is known that the signal conversion performance of an Efficient Trellis SDM is equal to or better than that of a full Trellis SDM, at a fraction of the computational cost. However, for the highest signal conversion performance a large Trellis order N should be used, which has a negative impact on the computational efficiency of the approach. In sec. 9.1 it will be shown that by altering the initial conditions of the system the test for uniqueness of the last N bits of the parallel solutions becomes redundant, and can be removed from the algorithm. The resulting algorithm is a practical realization of the pruned look-ahead algorithm with reuse. This algorithm with improved computational efficiency is formulated in sec. 9.2. Since no test for uniqueness is performed the Pruned Tree sigma-delta modulation algorithm requires more time to converge on a single solution than the Efficient Trellis sigma-delta modulation algorithm, and the possibility to reduce the required history length is investigated in sec. 9.3. Next, in sec. 9.4 the functional performance of the algorithm is investigated. Details related to an efficient implementation of the algorithm are discussed in sec. 9.5. Finally, conclusions are drawn in sec. 9.6.

9.1 Removing the test for uniqueness

The signal conversion performance of an Efficient Trellis SDM, typically, improves when the value of N is increased while the number of parallel paths M is kept constant. Although the number of solutions to evaluate

does not increase when N is selected larger, the required computational power does increase since longer bit sequences need to be compared. Especially in combination with a large value of M the test for uniqueness of the solutions is limiting the efficiency of the approach. Thus, in order to have a good signal conversion performance it is desirable to have the value of N large, but in order to have a computational efficient solution it is desirable to have N minimal. In this section it will be demonstrated that it is possible to adapt the Efficient Trellis sigma-delta modulation algorithm such that the signal conversion performance corresponding to a large value of N is realized, at the cost of $N = 0$.

In the Efficient Trellis sigma-delta modulation algorithm every clock cycle a test is performed to guarantee that all the parallel solutions that are maintained are unique in their newest N bits. Without this test it would be possible that the same feed-back solution would be investigated multiple times, resulting in a converter with, effectively, a reduced number of parallel paths. This test for uniqueness is performed continuously, i.e. the algorithm disallows that any two bitstreams are equal in their newest N bits. However, this is a much stronger constraint than the requirement that two bitstreams can not be identical. The difference between the two constraints is that in the last case the two bitstreams can still be equal over large fractions of their length, whereas in the first case the bitstreams can only be equal over lengths of at maximum $N-1$ symbols. Since the objective is to maintain M unique solutions at all times, it is clear that the selection constraint of the Efficient Trellis sigma-delta modulation algorithm is too strong and that it can be relaxed significantly. In fact, it is possible to guarantee that all the parallel solutions that are investigated are unique, at zero computational cost.

Consider the scenario of an Efficient Trellis SDM that does not check for uniqueness of the parallel paths and that maintains two parallel paths. At the start of the conversion process both loop filters are initialized the same, e.g. all integrator states are made equal to zero, and the path cost is initialized to zero. Now the two paths are both expanded with a '0' and a '1' symbol and the cost for appending those symbols is calculated and added to the running path cost. Since both loop filters have the same initial state, they will both react the same to the feed-back signals. As a result, the cost for appending the '0' symbol will be identical for both paths. The cost for appending the '1' symbol will be, in general, different from the cost for appending the '0' symbol, but again identical for the two paths. Thus, from the two paths originate four potential solutions with only two different cost values that are depending on the symbol that has been appended. Imagine that the cost for appending the '0' symbol is the lowest. If now the two paths with the lowest cost

are selected, the two paths that are resulting from appending the '0' symbol will be selected. The result of this selection is that two identical paths are passed to the next time step, and that the loop-filter states that are connected with these paths are also identical. Thus, already after the first conversion cycle the number of unique solutions has been reduced to one, and no look-ahead will be achieved.

In order to avoid the situation described above, the selection outcome of what solutions pass to the next clock cycle has to be altered, such that a solution with a '0' symbol and a '1' symbol is continuing. This can be realized efficiently by changing the initial conditions of the two paths. More specifically, the initial loop-filter states of the two paths need to be identical such that the different parallel solutions can converge, but the path cost can be initialized with different values, e.g. once with a zero value and once with a large value. In the example above with $M = 2$, such an initialization will result in four potential solutions with four different cost values. The four solutions are still consisting of two times two identical bit patterns, but two of these solutions are originating from the cheap path and two solutions are originating from the expensive path. As a result, the two solutions with the lowest cost will be both originating from the path with the initial running path cost of zero. Thus, a solution with a '0' symbol as the newest symbol and a solution with an appended '1' symbol are continuing to the next conversion cycle.

Because two different symbols are appended to the same solution, the loop-filter states of the two new paths will be different. As a result of the difference in the loop-filter states of the two continuing solutions, in the next conversion cycle the cost for appending the same symbol to the two solutions will be different, and four unique potential solutions with four different cost values will be resulting. Independent of what solutions are selected to continue, from now on the cost for appending the same symbol to both solutions will always be different since the paths have a different loop-filter state. Under the assumption that the loop filter is at least second order, i.e. has two integrator sections, it is impossible that the states of the filters will become equal again. Thus, from this point onwards the solutions that can be generated will all be unique and no additional test for uniqueness of the solutions is required.

In the example above it was illustrated that, with a proper initialization of the running cost values, no test for uniqueness of the solutions is required for a configuration with two parallel paths. Also in a setup with more parallel paths the test for uniqueness of the solutions can be omitted if the system is initialized properly. In this case all the M paths, except for one, need to be initialized with a high running path

cost. Because of this initialization, in the first conversion cycle the cheap path will for certain be extended with a '0' and a '1' symbol, resulting in two paths with a low cost value and $M-2$ paths with a high running path cost. In the next conversion cycle, the two cheap paths will be both extended with a '0' and a '1' symbol, such that four unique paths with a low running path cost result. This process repeats, until all the expensive paths are rejected by the system and are replaced with unique solutions. After $\lceil log_2(M) \rceil$ clock cycles the system is completely functional and all the M parallel paths will be active. Since all the loop filters receive the same input signal but different feed-back sequences, the loop-filter states will stay unique and all the solutions under investigation will be unique, without the necessity to check for uniqueness, as long as the loop filter is of at least second order. The case of a first order loop filter is not of interest and will not be considered, since the objective is to realize a high quality conversion quality.

9.2 Algorithm

The Efficient Trellis sigma-delta modulation algorithm without the test for uniqueness of the parallel solutions is, in principle, equal to the pruned look-ahead modulation algorithm with reuse of results, as described in sec. 6.2.3, but then practically realized. The Pruned Tree sigma-delta modulation algorithm [32] consists of an initialization phase, key for the algorithm to work, and a normal (operation) phase that is repeated for every input sample. The algorithm's performance and computational load is controlled by two parameters. The number of parallel paths is set by M, and the trace-back depth or latency of the algorithm is controlled by L.

9.2.1 Initialization phase

In the initialization phase, all the M parallel paths are setup for operation. This means that the loop-filter states of all the M parallel paths are reset to the same initial condition, e.g. all integrators states equal to zero. The initial condition of the running path cost values are not setup equal for all the paths, such that the system will start by diverging from a single state and that no test for uniqueness is required. This is accomplished by initializing the running path cost of exactly one path to zero, and the running path cost of the other $M-1$ paths to a large constant. The value of this constant should be at least equal to the running path cost that can be accumulated over $\lceil log_2(M) \rceil$ cycles when

non-matching feedback values are applied. As a result, during the first $\lceil log_2(M) \rceil$ conversion cycles all the paths that originate from the path that was initialized with a zero running path cost will be cheaper than the $M-1$ paths that were initialized with a high running path cost, and the system will automatically diverge from this single point.

9.2.2 Operation phase

The operation phase of the algorithm consists of six steps which are very similar to those of the Efficient Trellis sigma-delta modulation algorithm:

1. extend the M paths with a '0' and a '1' bit and calculate the $2M$ cost values;
2. calculate the $2M$ accumulated path cost values;
3. select the M paths with the lowest accumulated path metric and update the look-ahead filter states;
4. adjust the path metric values;
5. determine the output symbol;
6. invalidate the paths that have not converged.

In step 1 and 2 all the M paths are extended with the two possible symbols, and the resulting $2M$ accumulated path cost values are calculated. These two steps are identical to the first two steps of the Efficient Trellis sigma-delta modulation algorithm.

Step 3 is different from that of the Efficient Trellis sigma-delta modulation algorithm. Here the best M paths are selected, only based on their path metric and without testing for uniqueness of the newest N bits. The initialization phase of the algorithm ensures that all the parallel solutions that are investigated are different.

In order to keep the path metric values bounded, in step 4 the path metric values are adjusted by subtracting a value from them. By subtracting the cost of the cheapest path from all path metric values, it is guaranteed that the path metric values are always positive and minimal. Since the selection in step 3 is based on relative values of the path metric, this action can be performed without altering the conversion result.

In step 5 the output symbol is determined. This is realized by selecting the cheapest path and by determining the symbol that was appended to this solution L clock cycles ago.

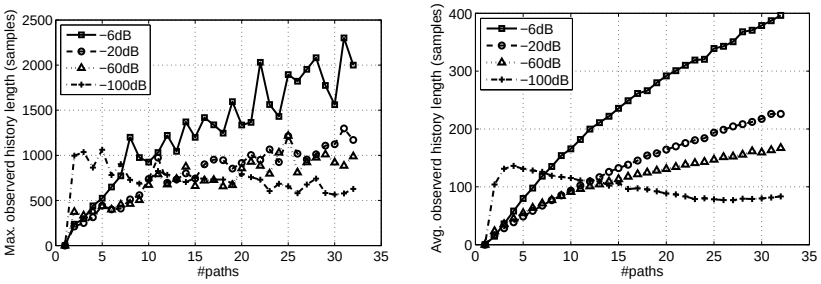
In step 6 a test is performed to determine if all the M remaining parallel solutions have converged on the same output symbol. If it is detected that a solution has not converged, the cost of the path is increased by a large amount, such that the path will not survive the next selection process. Effectively, this reduces the number of parallel solutions by one during the next conversion cycle, but this is not problematic since this should not occur very frequently. In order to minimize the amount of required memory and to maximize throughput, potentially at the cost of a slightly reduced conversion quality, it can also be desirable to use a history length L that is shorter than the length that is required to reach convergence under (nearly) all conditions. In this situation the test for convergence is certainly a necessity.

9.3 Required history length

Since the Pruned Tree sigma-delta modulation algorithm does allow long sequences of bits to be identical, it is likely that the algorithm will, typically, require more clock cycles to reach converge on the output symbol than the Efficient Trellis sigma-delta modulation algorithm. In order to verify this hypothesis, an experiment is performed in which, as a function of the number of parallel paths M, the required history length is determined for various input levels. This experiment is performed in a similar fashion as described in the previous chapters, i.e. at every clock cycle it is determined how far back in time all the paths become equal. On the basis of these findings the average and maximally required history length are calculated. The experiment is performed for loop-filter configuration SDM2, as described in app. B.

In fig. 9.1(a) the maximum required history length is plotted for values of M between 1 and 32. For small values of M, a relatively large maximum history length is observed for low input amplitudes. However, for values of $M \geq 8$ more clock cycles latency are required for large input signals than for low amplitude signals. The same behavior is also seen for the average required history length, depicted in fig. 9.1(b). Thus, the number of parallel paths M determines if it is easier to convert a high or a low amplitude signal.

In the situation of $M = 32$ a history length of at least 2400 samples is required, in order to avoid convergence problems. The average required history length is only 400 bits. In the case of an Efficient Trellis SDM with $M = 32$ and $N = 32$ a maximum required history length of approximately 1400 bits is observed for the same input signal, while there the average required history length is only around 275 bits. Clearly, the



(a) Maximum required history length. (b) Average required history length.

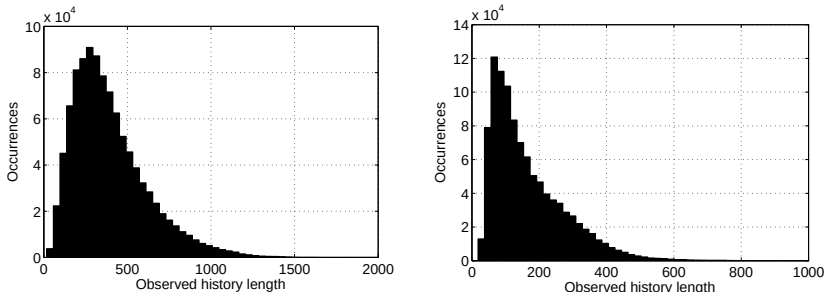
Figure 9.1: The maximum (a) and the average (b) observed required history length for loop-filter configuration SDM2 as a function of the number of parallel paths for various input levels.

maximum required history length has increased significantly more than the average value, which indicates that the Pruned Tree sigma-delta modulation algorithm investigates different solutions than the Efficient Trellis sigma-delta modulation.

Furthermore, the increased average and maximum observed history length suggest that potentially a higher conversion quality can be realized by the Pruned Tree sigma-delta modulation algorithm, since similar solutions are compared over a longer time span before a final selection is made. However, the use of such a large history length does have a negative influence on the computational throughput, and should only be considered when the absolute maximum obtainable performance is required. For the case of $M = 32$, in fig. 9.2(a) and fig.9.2(b), the distribution of the required history length for a 1 kHz input signal with an input amplitude of -6 dB and -60 dB is shown for SDM configuration SDM2. From these plots it is clear that only a minimal conversion performance penalty can be expected if a reduced history length of two to three times the maximum average observed history length is used instead, since only in very few cases the system will not reach convergence with all paths in this case.

If the system can reach convergence less often, this does not have to be a problem, as long as most of the paths reach convergence within the available history length. If a decision on the output symbol is forced, from all the parallel paths the best path will be selected, and the paths that did not converge will be stopped. This is not necessarily causing a significant reduction in conversion quality, since all the solutions under

investigation that are evaluated over many samples are describing the input signal very accurately. Thus, in order to realize a good computational efficiency, it seems very acceptable to use a much further reduced history length in combination with a check on convergence. The minimally required history length that does not cause a reduction in conversion performance is determined in the next section.



(a) $M = 32$ and $A = -6$ dB. (b) $M = 32$ and $A = -60$ dB.

Figure 9.2: Distribution of the observed history length for a -6 dB (a) and a -60 dB (b) 1 kHz input signal with $M = 32$ for SDM configuration SDM2.

9.4 Functional performance

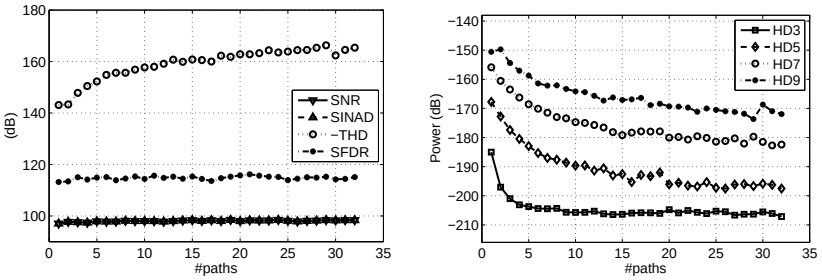
The functional performance of a Pruned Tree SDM is investigated on the basis of the classical signal quality indicators SNR, SINAD, THD, and SFDR. Next to these indicators, the SDM specific performance measures stability and noise modulation are investigated. Finally, a summary is given of the Pruned Tree SDM functional performance.

9.4.1 SNR, SINAD, THD and SFDR

From the previous chapters it is known that the potential performance improvement of a look-ahead modulator can be characterized by studying the conversion performance for two types of loop filters, i.e. one with and one without resonator sections. As before, loop-filter configuration SDM1 and SDM2 (see app. B) are studied for this purpose.

SDM1

In sec. 9.3 it was demonstrated the maximum required history length for a Pruned Tree SDM is significantly longer than that of an Efficient Trellis SDM. In the characterization of the performance of loop-filter configuration SDM1 a history length of $L = 4096$ is used, such that the system will always be able to realize convergence without invalidating paths. As a function of the number of parallel paths M the SNR, SINAD, THD, and SFDR are measured, as well as the power levels of the first four odd harmonics. The simulation length is equal to 1 million samples, and 128 coherent averages are performed in combination with 4 power averages in order to obtain reliable performance figures. The results of these measurements are depicted in fig. 9.3.



(a) SNR, SINAD, THD, SFDR. (b) HD3, HD5, HD7, HD9.

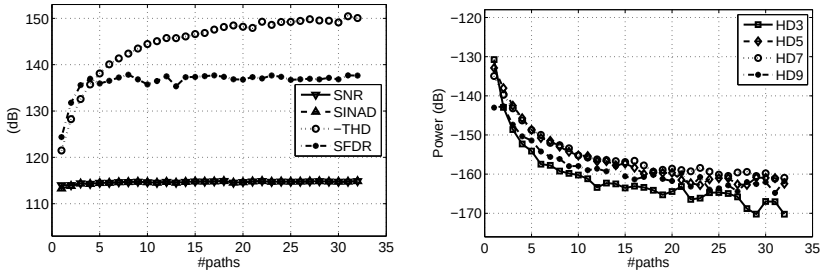
Figure 9.3: SNR, SINAD, THD, and SFDR performance for configuration SDM1 as a function of the number of parallel paths for a 1 kHz sine wave with a power of -6 dB (a) and the power in HD3, HD5, HD7, and HD9 (b) for a history length of $L = 4096$ samples.

The suppression of the harmonic distortion scales smoothly with the number of parallel paths M. By increasing the number of paths from one to 32 an improvement of the THD of 24 dB is realized. However, a comparison to the performance of an Efficient Trellis SDM with the same loop filter (fig. 8.7 and fig. 8.8) reveals that for small values of M there is no performance improvement realized compared to what is possible. More specifically, a comparison should be made to the best performance that can be realized by an Efficient Trellis SDM, and for a loop filter without resonator sections this is realized for small values of N when M is small. The performance that is realized by the Pruned Tree SDM is approximately equal to that realized by the Efficient Trellis SDM for N large, which can be 5 to 10 dB worse than the best values

that can be obtained. For the largest studied value of M, i.e. the case of $M = 32$, the THD performance of the Efficient Trellis SDM is still minimally better than that of the Pruned Tree SDM.

SDM2

With a setup equal to that as described for loop-filter configuration SDM1, the signal conversion performance for loop-filter configuration SDM2 is determined as a function of the number of parallel paths M. The experiment is performed for a history length of $L = 4096$ in order to maximize the possibility of reaching convergence with all paths. In fig. 9.4 the measurement results are shown.



(a) SNR, SINAD, THD, SFDR. (b) HD3, HD5, HD7, HD9.

Figure 9.4: SNR, SINAD, THD, and SFDR performance for configuration SDM2 as a function of the number of parallel paths for a 1 kHz sine wave with a power of -6 dB (a) and the power in HD3, HD5, HD7, and HD9 (b) for a history length of $L = 4096$ samples.

The power in the harmonic distortion components decreases steadily when the number of paths is increased, resulting in an THD improvement of 29 dB when 32 paths are used instead of one. A comparison to the performance of an Efficient Trellis SDM (sec. 8.5.1) shows that for loop-filter configuration SDM2 the performance of the Pruned Tree SDM is on a comparable level. For small values of M the performance of the Pruned Tree SDM is slightly better, for large values of M minimally worse. In order to realize the maximum obtainable improvement in SFDR only four parallel paths are required, whereas with an Efficient Trellis SDM eight parallel paths are required to reach this level. By using $M = 8$ an improvement of 20 dB in the THD is realized, which is also slightly better than what is realized by the Efficient Trellis SDM.

In order to verify the hypothesis that with only a limited impact on the signal conversion performance a shorter history length can be used, the SNR performance of loop-filter configuration SDM2 is characterized as a function of the history length L. The experiment is performed for $M = 8$, $M = 16$, and $M = 32$, with the history length L varying between 8 and 32768 samples. In order to get reliable SNR numbers one million samples are used and 8 power averages are performed. The results of the experiment are plotted in fig. 9.5.

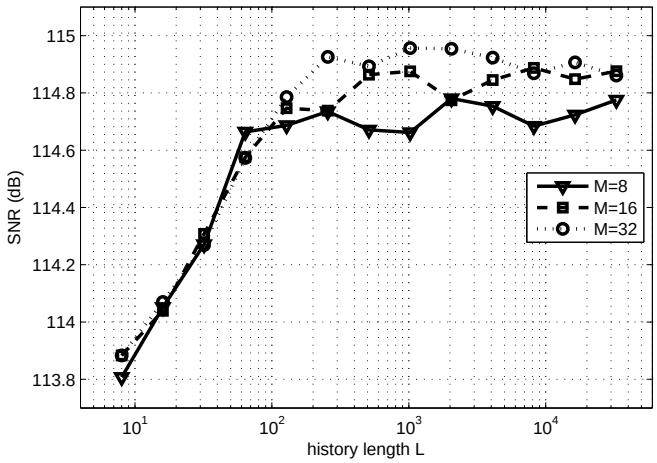


Figure 9.5: SNR performance for configuration SDM2 as a function of the history length L for a 1 kHz sine wave with a power of -6 dB.

From the figure it is clear that small values of the history length result in a significant decrease in the obtainable SNR, even in the case of $M = 8$. However, against expectation, also when very large values of the history length are used the SNR does not fully stabilize and variations in the order of 0.1 dB are recorded. Since in the experiment the input stimuli as well as the initial conditions are identical for each run, the resulting bitstreams will be identical if all the paths converge. Since the obtained SNR varies of function of the history length, it can only be concluded that the solutions do not always converge within the available history length, and that a decision on the outcome is forced. However, it is also clear that it is not required to use extremely long history lengths since the impact on performance is very limited. Depending on the number of parallel paths, a minimum history length is required to reach the full performance, and increasing the number further does not bring any

advantage. More specifically, in the case of $M = 8$ a history length of $L = 64$ samples seems sufficient to reach near optimal results. In a setup with $M = 16$ an approximate value of $L = 128$ is sufficient, and in the case of $M = 32$ around 256 samples of history length are necessary to reach the full SNR. From these results it can be concluded that the estimate that a history length of two to three times the average observed history length is required to reach good performance is too pessimistic. More specifically, a reasonable estimate of the necessary history length is in the order of 65% of the average observed history length, which coincides with the maximum of the histogram of the observed history length (fig. 9.2(a)). Thus, already when slightly more than half of all the solutions converge (near) optimal performance is reached.

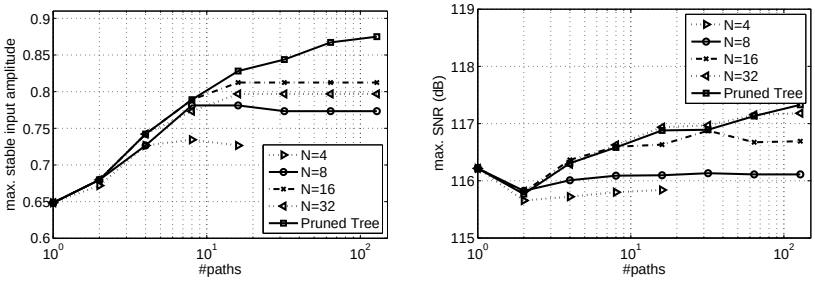
9.4.2 Converter stability

The stability of a Pruned Tree SDM, measured as the maximum input amplitude that can handled, and as the maximum loop-filter corner frequency that can be used with a given input signal, is investigated as a function of the number of parallel paths M.

Maximum stable input amplitude

From the SNR characterization experiments it is known that with a moderate history length very good performance levels can be achieved. Therefore, for the stability measurement a safe history length of $L = 1000$ samples, i.e. 4 times more than what is required according to the results obtained in the previous section, is used. The maximum amplitude of a 1 kHz sine wave that can be handled by a modulator with loop-filter configuration SDM2 (app. B) without becoming unstable is determined as a function of the number of parallel paths M. The results of this measurement are compared to the results obtained for an Efficient Trellis SDM, and are depicted in fig. 9.6(a).

In the case of the Efficient Trellis SDM the maximum stability is reached for a setup with $N = 16$ and at least 16 parallel paths. Increasing the number of parallel paths further does not improve the stability of the Efficient Trellis SDM. In the case of the Pruned Tree SDM the stability continues to improve when the number of parallel paths M is increased. It can be seen that the stability of the Pruned Tree SDM is always equal or better than that of the Efficient Trellis SDM when the same number of paths are used. If 128 parallel paths are used instead of one, the maximum input amplitude that can be handled increases from 0.65 to



(a) Stable input amplitude.

(b) Maximum SNR.

Figure 9.6: Maximum stable input amplitude (a) and the maximum SNR (b) as a function of the number of parallel paths for an Efficient Trellis SDM and a Pruned Tree SDM for configuration SDM2 for a 1 kHz sine wave.

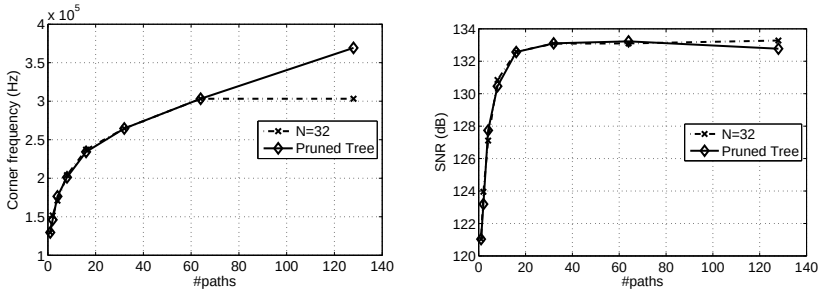
0.875. In the case of an Efficient Trellis SDM the maximum that can be handled is equal to 0.82. Although the difference in stability between the two modulator types seems negligible if it is expressed on a logarithmic scale, it can be very relevant in some specific cases. For example, for Super Audio CD mastering it is a necessity to support signals up to 0.714, and the increased stability of a Pruned Tree SDM will enable to encode such signals with more aggressive noise shaping than an Efficient Trellis SDM.

In the stability experiments in the previous chapters it was shown that the maximum SNR that can be realized is obtained for input levels that are below the maximum possible input level. In the case of a Pruned Tree SDM this is also the case. In fig. 9.6(b) the maximum SNR that can be obtained with a given number of parallel paths is plotted, confirming that the maximum SNR that is obtained is equal for an Efficient Trellis SDM and a Pruned Tree SDM. Thus, although a Pruned Tree SDM can achieve a larger stability than an Efficient Trellis SDM, it is not able to realize a higher peak SNR.

Maximum loop-filter corner frequency

As an alternative to the maximum input amplitude stability test, the stability that can be obtained by a Pruned Tree SDM is also characterized on the basis of the maximum corner frequency that can be used. The experiment is performed for a loop filter with resonator sections,

equal to those of loop-filter configuration SDM2 (app. B) and compared to the results obtained for an Efficient Trellis SDM with the same number of parallel paths and $N = 32$. The input signal that is used is, as before, a 1 kHz sine wave with an amplitude of -6 dB. The results of this measurement are shown in fig. 9.7(a).



(a) Stable corner frequency.

(b) SNR at max. corner freq.

Figure 9.7: Maximum loop-filter corner frequency that results in stable operation as a function of the number of parallel paths for an Efficient Trellis SDM with $N = 32$ and a Pruned Tree SDM for a -6 dB 1 kHz sine wave (a) and the associated SNR (b).

In the case of the Pruned Tree SDM, the obtainable corner frequency increases continuously with the number of parallel paths. For 64 or less parallel paths, exactly the same maximum obtainable corner frequency is found for the Pruned Tree SDM as for the Efficient Trellis SDM. Unlike the Efficient Trellis SDM the stability of the Pruned Tree SDM improves further when the number of paths is increased from 64 to 128, resulting in a higher maximum corner frequency. As already demonstrated in the same experiment in the previous chapters, the SNR does not continue to improve when higher corner frequencies are realized. From approximately 250 kHz onwards the obtained SNR is virtually constant at a level of 133 dB, as shown in fig. 9.7(b). The Efficient Trellis SDM and the Pruned Tree SDM realize exactly the same SNR when the same corner frequency is used. For the case of 128 parallel paths, a higher corner frequency is achieved by the Pruned Tree SDM, which results in a minimally lower SNR than what is achieved for the lower corner frequency by the Efficient Trellis SDM, caused by the reduced stability that is resulting from the higher corner frequency. Thus, although the Pruned Tree SDM is able to handle more aggressive noise shaping, the usage of this higher corner frequency does not result in a higher SNR. An analysis of the phenomenon that the SNR is limited, and can not be increased

by selecting a higher corner frequency, is performed in chapter 12.

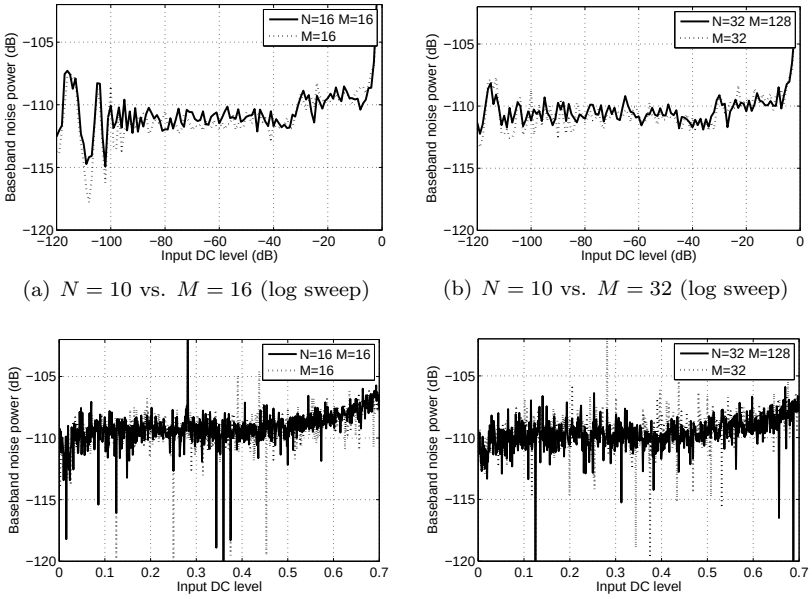
9.4.3 Noise modulation

From the previous chapter it is known that the global amount of noise modulation of an Efficient Trellis SDM is equal to that of a full Trellis SDM. If rational DC levels are applied, the Efficient Trellis SDM shows a slightly reduced performance, because it is able to realize lower noise levels for some specific DC levels. However, this is not a major problem since typical input signals are not DC, and it is only the global variation of the in-band noise that is relevant. Since the computational load of an Efficient Trellis SDM is significantly less than that of a full Trellis SDM, the performance of an Efficient Trellis SDM is used as a reference point to evaluate the in-band noise modulation performance of a Pruned Tree SDM.

For loop-filter configuration SDM1 (see app. B) the amount of noise present in the 0-20 kHz band for various DC input levels is measured. Two experiments are performed, i.e. one with a logarithmic selection of the DC levels and one with a linear selection. The experiment with the logarithmic selection is the most relevant, and is used to illustrate how the amount of in-band quantization noise varies globally as a function of the amplitude of the input signal. The linear amplitude selection criterion is such that rational DC levels are resulting, which will often trigger limit cycles in a typical SDM and that can result in an increase or decrease of the noise level. As mentioned before, the outcome of this experiment is less relevant for the final performance evaluation, but it can still give insight in the encoding quality of the modulator.

In fig. 9.8(a), for input levels between -120 dB and -3 dB, the in-band noise level is plotted for an Efficient Trellis SDM with $N = 16, M = 16$ and compared to the in-band noise level of a Pruned Tree SDM with $M = 16$. For input levels below -90 dB the performance of the Pruned Tree SDM is slightly worse than that of the Efficient Trellis SDM. For larger input levels the two modulators realize the same in-band noise level. When rational DC levels are supplied to the two modulators (fig. 9.8(c)), similar performance levels are obtained. Thus, except for very small input levels, the noise modulation performance of the two modulators is equal.

The same comparison is also made for an Efficient Trellis SDM with $N = 32, M = 128$ and a Pruned Tree SDM with $M = 32$. The global noise modulation (fig. 9.8(b)) of the two modulators is equal. In the case of rational DC input levels a difference in the in-band noise level



(a) $N = 10$ vs. $M = 16$ (log sweep)

(b) $N = 10$ vs. $M = 32$ (log sweep)

(c) $N = 10$ vs. $M = 16$ (linear sweep)

(d) $N = 10$ vs. $M = 32$ (linear sweep)

Figure 9.8: In-band noise as a function of the DC input level. Comparison of an Efficient Trellis SDM with $N = 16, M = 16$ to a Pruned Tree SDM with $M = 16$ in fig. (a) for a logarithmic input level selection and in fig. (c) for a linear input level selection. In fig. (b) and fig. (d) the comparison of an Efficient Trellis SDM with $N = 32, M = 128$ to a Pruned Tree SDM with $M = 32$ for a logarithmic, respectively, linear input level selection.

can be observed between the two modulators, as illustrated in fig. 9.8(d). More specifically, although for both modulators the in-band noise power is around -110 dB for inputs up to a level of 0.5, the Pruned Tree SDM has more levels for which a much higher or lower noise level is realized. Thus, for some DC levels a higher than average in-band noise floor is realized, whereas for other DC levels a much lower noise floor is obtained. Therefore, on the basis of the experiment, it can be concluded that the global variation of the in-band noise power of a Pruned Tree SDM with 32 parallel paths is on the same performance level as that of an Efficient Trellis SDM with 128 parallel paths. However, if also rational DC levels are considered, the quality that is obtained by the Pruned Tree SDM with 32 parallel paths is on a slightly lower level than that of the Efficient Trellis SDM with 128 parallel paths.

9.4.4 Summary

Independent of the type of loop filter, i.e. with or without resonator sections, the THD generated by a Pruned Tree SDM reduces when the number of parallel paths is increased. The largest improvement is realized by increasing the number of paths from one to eight, but also increasing the number of parallel paths further keeps improving the THD performance. The realization of improvements in the SFDR depends on the presence of resonator sections.

In the case of a loop filter without resonator sections, the signal conversion performance of a Pruned Tree SDM with M parallel paths is equal to, or in some conditions slightly lower than, what can be achieved with an Efficient Trellis SDM when the best value of N is selected for the given M. More specifically, when a relatively small number of parallel paths is used, i.e. up to approximately 30, an Efficient Trellis SDM realizes its best performance for a value of approximately $N = 16$. In the case of a Pruned Tree SDM, since effectively N is equal to infinity, a slightly lower improvement in the THD is realized for limited values of M. If M is increased to large values, the performance of an Efficient Trellis SDM and Pruned Tree SDM become equal, since the best performance of the Efficient Trellis SDM will now be realized for a large value of N, i.e. a situation similar to that of the Pruned Tree SDM.

A comparison of the performance increase realized for a loop filter with resonator sections, shows that the improvements that can be realized by an Efficient Trellis SDM and a Pruned Tree SDM are very similar. For a low number of parallel paths the Pruned Tree SDM is more effective, while for a large number of parallel paths the Efficient Trellis SDM is

minimally more effective. For example, in the case of loop-filter configuration SDM2 only four parallel paths are required by the Pruned Tree SDM in order to suppress all harmonic distortion to below the quantization noise floor. If an Efficient Trellis SDM is used, eight parallel paths are required to reach the same performance. On the other hand, if 32 parallel paths are used, the Efficient Trellis SDM realizes a THD which is 2 dB better than what is realized by the Pruned Tree SDM.

In sec. 9.3 it was shown that it can take thousands of clock cycles before a Pruned Tree SDM reaches convergence on the output solution. However, an investigation of the impact of the actual used history length on the signal conversion performance has shown that a very limited history length is sufficient to realize the maximum possible performance. More specifically, if a history length of at least 65% of the average observed history length is used, virtually no impact on signal conversion performance can be detected. In the case of 32 parallel paths this means that a history length of 256 samples is sufficient, whereas the maximum observed history length is around 2500 samples.

The maximum input amplitude a Pruned Tree SDM can handle without becoming unstable is a direct function of the number of parallel paths, i.e. a larger number of parallel paths results in a larger input amplitude that can be converted. Comparison with an Efficient Trellis SDM reveals that for an equal number of parallel paths the stability of a Pruned Tree SDM is always equal or better. In the case of an Efficient Trellis SDM with loop-filter configuration SDM2 the maximum amplitude that can be converted is 0.81 instead of the default value of 0.65 for one path. This value is realized for $M = 16$, and increasing the number of paths further does not result in a larger amplitude that can be converted. In the case of a Pruned Tree SDM the maximum amplitude can be increased to 0.88 when 128 parallel paths are used.

If the stability of a converter is expressed as the maximum loop-filter corner frequency that can be used with a 1 kHz -6 dB input signal, a similar result as described above is obtained. In this case, the maximum stability of the Efficient Trellis SDM is realized when 64 parallel paths are used, and the highest useable corner frequency is 300 kHz. The maximum useable corner frequency for the Pruned Tree SDM is equal to that of the Efficient Trellis SDM when 64 or less paths are used. However, if the number of paths is increased to 128 the maximum corner frequency that can be used still increases further to 370 kHz. Thus, a Pruned Tree SDM is as stable as an Efficient Trellis SDM for a low number of paths, but provides better scalability that results in a larger stability when many parallel paths are used.

The amount of in-band noise modulation of a Pruned Tree SDM is comparable to that of an Efficient Trellis SDM when a limited number of parallel paths is used. When 16 parallel paths are used, for DC levels below -90 dB the Pruned Tree SDM shows slightly more variation in the in-band noise, but for larger DC levels the two modulators realize an equal performance. In the case of an Efficient Trellis SDM 128 parallel paths are required in order to realize the same performance as a full Trellis SDM with $N = 10$. However, a Pruned Tree SDM with only 32 parallel paths realizes approximately the same amount of global in-band noise modulation as an Efficient Trellis SDM with 128 parallel paths. If rational DC levels are also considered the performance of the Efficient Trellis SDM is slightly better than that of the Pruned Tree SDM.

9.5 Implementation aspects

The Pruned Tree sigma-delta modulation algorithm consists of less complex steps than the Efficient Trellis sigma-delta modulation algorithm. As a result, it is easier to realize an efficient implementation of the Pruned Tree sigma-delta modulation algorithm. More specifically, since there is no check for uniqueness of the parallel paths required, the selection step in which the best M solutions are selected can become much more efficient. Although the initialization phase is crucial for the correct functioning of the algorithm, i.e. only with a proper initialization the check for uniqueness can be omitted, it is not interesting from an implementation point of view since it is only executed once and has no influence on the computational efficiency.

In the previous chapter it was concluded that to realize an efficient implementation of an Efficient Trellis SDM it is important to have an efficient implementation of the look-ahead loop filter such that no unnecessary calculations are performed, and to realize the output symbol selection by storing the history of each path directly with its look-ahead filter and reading out a memory location directly in order to avoid a trace-back action. These points also hold for the efficient realization of a Pruned Tree SDM. Furthermore, in order to limit the required history length, it is necessary to implement a test for convergence of the parallel paths. This can be realized in an efficient way by combining it with the output symbol selection process, as discussed in sec. 7.6.3 for the traditional Trellis sigma-delta modulation algorithm.

In the Efficient Trellis sigma-delta modulation algorithm the selection step is the most time consuming part of the algorithm. This is caused by the test for uniqueness, which limits the complexity of the selection

operation to $\Omega(M^2)$, as shown in sec. 8.6.1. However, in that same section it is demonstrated that it is possible to reduce the complexity of the cost sorting problem by implementing a modified insertion sort and processing all the paths in sorted order. As a result of that approach, the computational complexity of the sorting becomes, on average, almost linear with the number of paths, i.e. $\Omega(2M)$.

In the Pruned Tree sigma-delta modulation algorithm the selection step only consists of determining, from the $2M$ available paths, the M paths with the lowest cost. As a result, this problem can be most efficiently solved by implementing the modified insertion sort and processing all the paths in the sorted order, as described in sec. 8.6.1. Because of the limited number of parallel paths, no other sorting technique will be faster.

In summary, by combining the implementation knowledge from the previous chapters, it is possible to implement the Pruned Tree sigma-delta modulation algorithm such that the computational load scales in an almost linear fashion with the number of parallel paths. However, note that if the number of paths is increased, also the history length L should be increased in order to obtain the maximum performance. Such an increase can have a negative impact on the throughput, since more data needs to be copied when a path is duplicated if it continues to the next conversion cycle with both of the two possible symbols appended. Depending on the memory access model, the impact of the increase of the history length can be significant, or not visible at all.

In order to illustrate that indeed the processing time is virtually linear with the number of parallel paths, in fig. 9.9, as a function of the number of parallel paths, the time required to encode 524 million samples is plotted. Clearly, the required time increases in a linear fashion with the number of parallel paths. All the calculations were performed on a single 3.0 GHz CPU. The plot shown is obtained for a history length of 1024 samples, although for a history length of 512 samples as well as 4096 samples the same processing time is recorded.

9.6 Conclusions

An essential element of the Efficient Trellis sigma-delta modulation algorithm is the test for uniqueness of the parallel solutions. However, this test is very computational intensive, especially for large values of M and N, and reduces the efficiency of the algorithm to $\Omega(M^2)$. A second observation is that the signal conversion performance of an Efficient Trellis

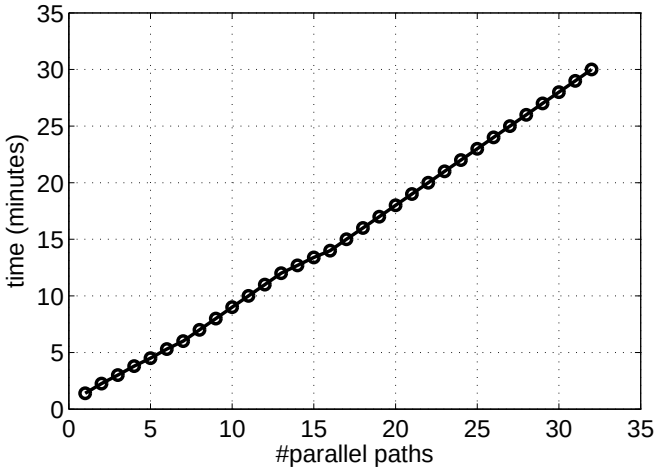


Figure 9.9: Time required to encode 524 million samples as a function of the number of parallel paths. The history length used was 1024 samples. All the computations were performed on a single 3.0 GHz CPU.

SDM, typically, improves when N is increased. Thus, in order to maximize conversion quality it is desirable to have N large, but this reduces the throughput of the converter. By introducing a proper initialization step in the Efficient Trellis sigma-delta modulation algorithm, the test for uniqueness of the parallel solutions can be removed, and a computationally more efficient converter is resulting that is akin to an Efficient Trellis SDM with N equal to L. The resulting solution is referred to as a Pruned Tree SDM, and is a practical realization of a pruned look-ahead modulator with reuse of results, as introduced in sec. 6.2.3.

Since no test for uniqueness is performed in a Pruned Tree SDM, the computational complexity of the algorithm is controlled with only two parameters. The number of parallel paths is specified by M, and the history trace-back length is specified with L. Except for the absence of the testing for uniqueness of the selected solutions, the algorithmic steps of the Pruned Tree sigma-delta modulation algorithm are equal to that of the Efficient Trellis SDM. However, before the algorithm can operate on input data, a proper initialization is required. In this initialization step the running path cost of all the parallel solutions, except one, has to be set to a very large value. As a result of this startup condition, all the parallel solutions will initially diverge from one solution, before they

can converge on the final selected solution. After the divergence phase all the loop filters will have a different internal state, and it is not be possible anymore that the same solution is investigated twice.

A result of the absence of testing for uniqueness of the parallel solutions is that it can take thousands of conversion cycles before all the parallel solutions converge on a single output bit. Thus, an equally large value of L would be required. However, running a converter with such a large value of L will have a negative impact on the throughput, and is not desirable. By testing for convergence of the solutions, which can be implemented with minimal overhead by combining it with the output symbol selection step, it is possible to operate the converter with a smaller value of L. It has been shown that virtually no signal conversion performance loss is introduced if the history length is selected as at least 65% of the average observed history length. This reflects to a value of about 10% of the maximum observed history length. For example, for a converter with $M = 32$ a history length of $L = 256$ samples is sufficient while the maximum observed history length is 2500 samples.

The signal conversion performance of a Pruned Tree SDM has been investigated and compared to that of an Efficient Trellis SDM. Similar to the operation of an Efficient Trellis SDM, the performance of a Pruned Tree SDM increases when the number of parallel paths is increased. However, there are some subtle differences in the achieved performance. In the case of a loop filter without resonator sections, the performance of a Pruned Tree SDM is, for small values of M, typically, slightly less than what can be achieved with an Efficient Trellis SDM if the optimal value of N is used for the selected value of M. If a large number of parallel paths is used the optimal value of N will be large, and the performance of the two converters becomes equal. If a loop filter with resonator sections is used, the performance of a Pruned Tree SDM is, for values of M up to 16, slightly better than that of an Efficient Trellis SDM. If M is selected larger, the performance of the Efficient Trellis SDM becomes minimally better.

Compared to an Efficient Trellis SDM, the stability realized by a Pruned Tree SDM is always equal or better. More specifically, for small values of M the two converters realize the same stability, but for larger values of M the stability of the Pruned Tree SDM keeps increasing while the Efficient Trellis SDM does not become more stable when more than 16 to 32 paths are used.

The amount of noise modulation of a Pruned Tree SDM and an Efficient Trellis SDM is on a comparable level. For $M = 16$ the Efficient Trellis SDM performs minimally better. However, in order to realize the same

performance as a full Trellis SDM with $N = 10$ an Efficient Trellis SDM requires 128 parallel paths, whereas a Pruned Tree SDM with only 32 paths realizes virtually the same performance.

In sec. 9.5 the implementation challenges of a Pruned Tree SDM have been investigated. It has been demonstrated that, by combining the implementation knowledge from the previous chapters, it is possible to realize an implementation of which the computational complexity scales linearly with the number of parallel paths. This enables an easy to assess tradeoff between the required processing time and the resulting conversion quality.

In general, it can be concluded that the overall conversion performance of a Pruned Tree SDM is approximately equal to that of an Efficient Trellis SDM, but realized at a much reduced computational load. In the specific case of a modulator with a loop filter without resonator sections the obtainable improvement in signal conversion performance is slightly less. However, in the much more typical case of a modulator with a loop filter with resonator sections, the signal conversion performance of a Pruned Tree SDM is better than that of an Efficient Trellis SDM for $M \leq 16$. The situation of $M \leq 16$ is also the most interesting one, since the biggest improvement in signal conversion performance is realized when M is increased from one to eight. In practice, a good choice would be to use between four and eight paths, since this can bring a significant increase in SNR and/or input range and will reduce the harmonic distortion components to levels below the quantization noise floor, at a limited increase of the computational load.

Chapter 10

Pruned Tree sigma-delta modulation for SA-CD

In the previous chapters it was demonstrated that it is possible to improve the signal conversion performance of an SDM significantly by applying look-ahead techniques. These improvements include the realization of a higher SNR and support for a larger signal swing. From all the techniques described the Pruned Tree sigma-delta modulation algorithm is the most computationally efficient and is able to realize the largest improvements. This technique is therefore, in principle, ideal for realizing high quality Super Audio CD (SA-CD) encodings. However, for a modulator to be usable for the creation of SA-CD content, it needs to fulfill a number of specific requirements, which are discussed in sec. 10.1. An outcome of this discussion is that all the requirements, except one, can be fulfilled by any of the previously discussed look-ahead modulators. The problem that is not solved by any of the earlier described modulators is the inability to generate bitstreams that are compatible with the SA-CD lossless data compression algorithm. In order to come to a look-ahead solution that is also taking the lossless SA-CD data compression into account, the basics of the SA-CD lossless data compression algorithm are fist discussed in sec. 10.2. Then, in sec. 10.3, the dual optimization criterion is introduced. The resulting algorithm is presented in sec. 10.4. The functional performance of the solution, including the compression gain and the signal conversion performance, is studied in sec. 10.5. The important aspects for an efficient implementation of the algorithm follow in sec. 10.6. Finally, conclusions are drawn in sec. 10.7.

10.1 Requirements of an SA-CD modulator

In order for a modulator to be suitable for SA-CD encoding purposes, a number of constraints have to be fulfilled. Some of these constraints are hard constraints, defined in the annexes of the SA-CD standard documentation [47], and some are soft constraints.

For example, in annex D.4 of the SA-CD standard the maximum allowed amount of mid to high frequency noise power is described. If the output of a modulator contains more noise power in this specific frequency region, its output bit-stream will not pass the compliance test and can not be used as SA-CD content. By proper design of the modulator's loop filter the shape of the noise-shaped spectrum can be controlled and it is possible to guarantee that the high frequency noise power is always low enough.

Another constraint imposed by the SA-CD standard is the maximum allowed (low frequency) signal amplitude, which is defined in annex D.3 of the Scarlet book. Although from a signal conversion perspective it is, typically, not desirable to use the maximum signal amplitude, i.e. the maximum SNR is in most cases obtained for lower amplitudes, from a commercial point of view it is desirable to have the maximum loudness of a signal, where loudness is maximized by first compressing the signal (reducing the dynamic range) and then increasing the signal amplitude to the maximum level possible. The reason to have a maximum loudness is two-fold. Firstly, a signal with more loudness contains more power and is easier to detect by a car radio, which is necessary to reach as many potential listeners as possible. Secondly, if two similar signals (music records) are compared, the one that has more loudness is remembered better and is perceived to be of a higher quality. Thus, although a maximization of the loudness reduces the signal quality and destroys the musical fidelity, it leads to more sales. Since music recordings are made to be sold and earn a living from, it is a necessity that a modulator can handle the maximum allowed signal swing of +3.1 dB SACD[1], which translates to an amplitude of 0.714. For a traditional SDM that realizes an SNR of approximately 120 dB, the magic number that is used in marketing campaigns to illustrate the quality of the SA-CD format but that is not enforced by the standard, it is difficult to handle the maximum allowed signal level without becoming unstable. The typical solution for this problem is to add clippers to the modulator that prevent

[1] a sine wave with a peak amplitude of 50% of the theoretical maximum has a level of 0 dB SACD, see annex D.2 of the Scarlet book. Normalized to a feedback value of ±1.0 this reflects to an amplitude of 0.5

it from becoming unstable [46,52,53], but that reduces the signal encoding quality significantly when activated. Alternatively, a loop filter with less aggressive noise shaping is sometimes used, such that the modulator becomes more stable and can handle larger input signals, at the cost of a reduced SNR. Application of a look-ahead modulator can provide, at the cost of an increase of the computational load, a more elegant solution to this problem and can combine a high SNR with support for a large signal swing.

Besides the constraints described above, there is another constraint that can cause significant problems in the production of SA-CD content. In order to fit 74 minutes of audio data on a 4.7 gigabyte disc, both in a six-channel surround format and a stereo format, the SA-CD standard relies on lossless data compression. Lossless data compression does not alter the binary data, i.e. the result of compressing and decompressing data is identical to the original input data. Lossy data compression, on the other hand, does alter the binary description, in such a way that the impact on the perceived quality is minimal. Examples of lossy audio data compression are the MP3 format and the audio signal on DVD and Blu-ray video discs. In the case of lossy data compression, the reduction of the required storage space, called the compression ratio or the compression gain, can be selected upfront. Obviously, a higher compression ratio will not only reduce the amount of required data storage, but also the remaining audio quality. In the case of lossless data compression, the compression ratio can not be selected since it is a function of the data. Thus, the amount of required storage is variable and not known upfront. Since the amount of storage space on an SA-CD disc is fixed, it is clear that a minimum average compression ratio is required in order to fit all the compressed data on a disc. However, this minimum average compression ratio can not be guaranteed since it is a function of the 1-bit encoded data, that is a function of the audio signal and the encoding SDM. In typical situations no problems occur, but cases are known where it was not possible to fit all the data on a disc. Clearly, there is a desire to improve the compression ratio in order to avoid such situations. Since the audio signal can not be changed, the only two possibilities to influence the compression ratio are the lossless data compression algorithm and the SDM. The possibilities to improve the lossless data compression algorithm are limited since the decoder is fixed by the standard, which leaves only the possibility to improve the encoder side. However, extensive research on this topic has not resulted in significant improvements. The basic influence of the SDM on the compression ratio is understood, i.e. a modulator that realizes a higher SNR produces data that is more difficult to compress, but also here the attempts to

realize improvements have failed. Fortunately, application of look-ahead techniques can enable a modulator that realizes a high SNR and that produces data that is easy to compress.

10.2 SA-CD lossless data compression

The intention of this section is to give a global overview of how the Direct Stream Transfer (DST) algorithm, the lossless data encoding algorithm that is part of the SA-CD standard, realizes a highly compressed data stream that is easy to decompress by an SA-CD player. The purpose is not to explain all the details of the algorithm. For a rather elaborate description please refer to [28, 34, 39] or to the SA-CD standard documents [47].

The 1-bit signal, called Direct Stream Digital (DSD) in the SA-CD terminology, is compressed by the DST encoding algorithm using a three-step approach. In the first step framing is performed, the second step consists of prediction filtering, and in the third step arithmetic encoding is performed on the output of the second step. On the decoder side the steps are repeated in the reverse order, as schematically indicated in fig. 10.1. The details of the decoding steps of the algorithm are not very interesting, but it is insightful to understand the encoding process since this contains the key to realize a look-ahead modulator that is generating data that can be compressed well.

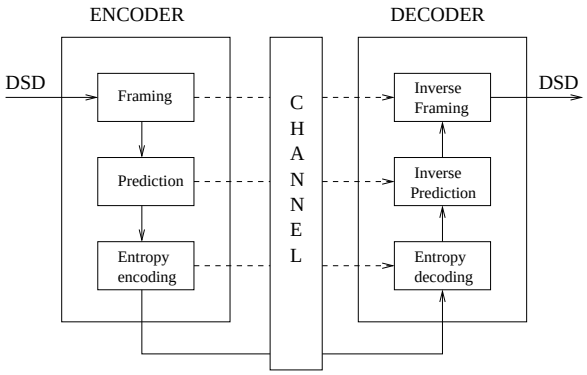


Figure 10.1: Schematic overview of the DST codec. Dashed lines indicate side data, thick lines the main signal flow.

In the first step of the algorithm the continuous stream of input data

is divided into small blocks of data, called frames. Each frame consists of 37 632 bits, which corresponds to 1/75 of a second. The purpose of the framing is two-fold. Firstly, the framing is necessary to provide easy "random" access to the audio data during playback. For the same reason, each frame needs to be independently encoded, which enables the player to decode separate frames without knowledge about preceding frames. Secondly, because of the framing the audio content in a frame can be regarded as (quasi-) stationary, which is the underlying assumption of the prediction process. The framing rate is chosen such that the assumption of quasi-stationary audio is reasonable, while it still does not result in excessive overhead.

In the second encoding step a FIR prediction filter, with a length of up to 128 taps, is derived for the frame. This prediction filter is then applied to the block of 1-bit input data (signal b in fig. 10.2) to generate a second 1-bit sequence q, i.e. the 1-bit input data b is filtered with the prediction filter and the floating point output z of the prediction filter is quantized to give a new 1-bit sequence q. If the prediction filter is perfect, the predicted sequence is equal to the input data. Because of the limited length of the prediction filter, typically, there will be some errors in the predicted bitstream. By taking the difference between the input sequence b and the output sequence q the prediction errors e can be found. Since both b and q are 1-bit signals the difference can be calculated by performing a 1-bit addition, as depicted in the figure. It is clear that if the prediction errors and the prediction filter coefficients are available, it is possible to reconstruct the original input signal.

If in the second step a proper prediction of the input signal is made, the error signal that is derived consists mainly out of '0' bits, and only on the locations where the prediction was incorrect '1' bits. Instead of storing this signal directly, in the third step of the algorithm, arithmetic encoding is performed on this signal. The arithmetic encoder operates on the error signal e and the result of of the probability table lookup (signal p) to generate the encoded signal d. The result of this operation is that the amount of data that is required to describe the input signal, i.e. signals d, t, and h in fig. 10.2, is reduced to nearly the theoretical minimum. If less prediction errors are made the amount of data to store reduces, and if more prediction errors are made more bits are required to store the signal.

Finally, the combination of the prediction coefficients h and the arithmetic encoded error signal d with the accompanying probability table t, is stored on the disc. If the combination of these signals requires less bits than the original input signal the procedure was successful and a com-

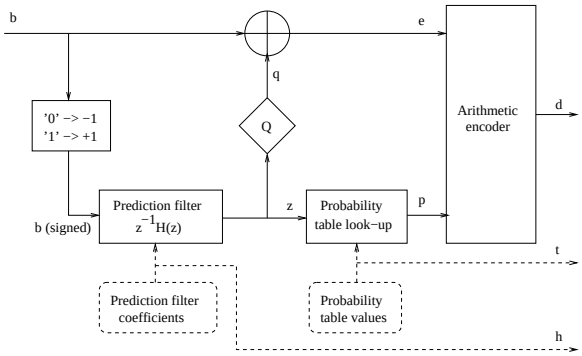


Figure 10.2: Schematic overview of the DST encoder. The input bits b come from the framing step. The output consists of the arithmetically encoded data d, probability table values t, and prediction filter coefficients h.

pression ratio larger than one has been realized. It is clear that there are two mechanisms that have an influence on the achievable compression ratio. More specifically, the prediction coefficients require storage space and the encoded error signal needs to be stored. If less prediction coefficients are used, the space required to store them decreases. However, if this results in more prediction errors, the error signal will require more space. The encoding algorithm attempts to find the optimal number of prediction coefficients, such that the final combination requires the least amount of space.

From the above description it is clear that the compression ratio will be high when the input signal is easy to predict, and that the compression ratio will be low when the input is difficult to predict. This makes sense, since from information theory we know that if a signal is difficult to predict its entropy is high, and more bits are required to store the signal. In the case of lossless data compression for SA-CD, the signal to compress is the 1-bit output of an SDM. The information in the 1-bit signal is consisting of the audio signal and the quantization noise, which is also partly a function of the audio signal. Although the low frequency audio signal is relatively easy to predict, the DST prediction filter, typically, attempts to predict the high frequency quantization noise, since this represents most of the signal power and will result in the minimum number of prediction errors. To illustrate this, in fig. 10.3 an example SDM output spectrum is shown in combination with the predicted spectrum. In this example the prediction filter contains 128

filter taps. Clearly, the prediction filter attempts to describe the high frequency tones accurately, while the low frequency input signal is only approximated very coarsely.

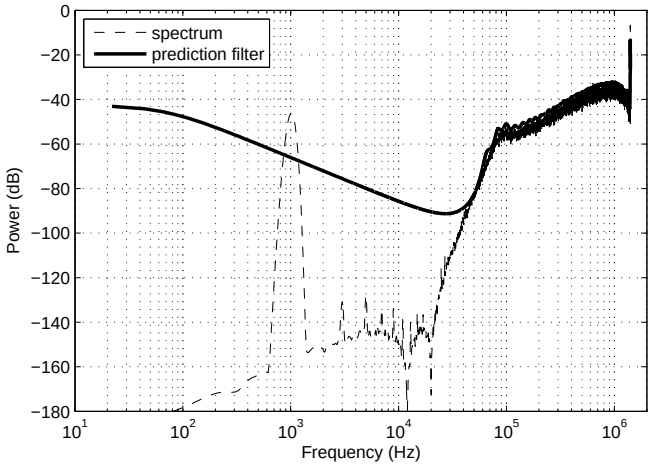


Figure 10.3: Example SDM output spectrum and the 128 taps prediction filter transfer.

From the above it can be understood why modulators with an aggressive noise-shaping characteristic, typically, generate bitstreams that can be compressed less than modulators with mild noise-shaping loop filters. More specifically, if a loop filter is more aggressive, i.e. it has a higher corner frequency or the filter order is higher, the SNR is higher and there is less correlation in the output bitstream. Because of this, the high frequency quantization spectrum becomes more noise alike and is less predictable. If a loop filter is less aggressive, typically, more distortion is added in the low frequency region and strong tonal behavior is present in the high frequency region. As a result, there is more redundancy in the signal and it is easier to predict. Thus, if the high frequency spectrum of a modulator can be made more predictable, the average compression ratio of the output bitstream will become higher. However, in the case of a normal SDM, the high frequency spectrum is correlated with the low frequency input signal [50] and can not be altered without introducing coding errors. In the case of a look-ahead modulator, because of the increased stability and the improved encoding properties, it is possible to increase the correlation in the high frequency region without adding correlation to the low frequency part. This can

be achieved by performing a dual optimization that takes both the signal and the predictability of the output bitstream into account. As a result of this approach it is possible to generate a bitstream that can be compressed better, without jeopardizing the signal quality.

10.3 Dual optimization

The look-ahead modulators discussed in the previous chapters are all using the same optimization criterion, i.e. they minimize a single cost function, where the cost function is defined as the energy at the output of the loop filter. This criterion results in a high quality signal encoding with reduced distortion, compared to a normal SDM. Furthermore, the criterion provides a stabilizing effect on the modulator, enabling more aggressive noise shaping or larger signal amplitudes. However, with this optimization criterion no attention is being paid to the predictability of the signal. In order to generate a signal that is easier to compress, a second optimization criterion that measures the predictability of the signal will be added. Since the predictability is measured on the 1-bit output signal of the modulator and is not taking the output of the first cost function that measures the signal quality into account, a second cost function is added in parallel to the first, as proposed in sec. 4.4. For readability, the structure of the generic noise-shaping quantizer with two parallel cost functions from sec. 4.4 is repeated in fig. 10.4.

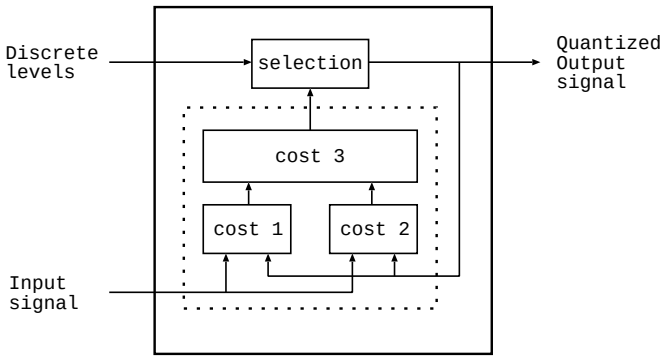


Figure 10.4: Noise-shaping quantizer with two parallel cost functions, repeated from fig. 4.7.

A third cost function combines the output of the first two cost functions, in such a way that a good balance between the achieved signal

quality and the predictability is achieved. More specifically, the signal quality should ideally not reduce because of the output symbols that are preferred from a predictability perspective. Since for large input signals there are few encoding possibilities, the second cost function should not be dominant for these signals. However, for small signals there is much more freedom in selecting bit sequences, and here the predictability of symbols should be taken into account.

10.3.1 Predictor cost function

Ideally, the prediction cost function of the look-ahead modulator should evaluate the predictability of the output bitstream using the same prediction filter as the one used during the DST encoding. However, in the DST encoding algorithm the prediction filter is derived for a complete frame of output bits. Since a frame consists of 37 632 bits, also in the case of a look-ahead modulator with a very large look-ahead depth, this data is not all available and will only be generated by the modulator in the future. Therefore, it is, by definition, impossible to optimize the output of a modulator for the actual DST prediction filter, since such an optimization would alter the output bitstream, which would then again result in a different prediction filter. In addition to this there is a practical issue, since the derivation of the prediction filter is a very computational intensive process. Thus, embedding the actual, or an approximation to the actual, prediction filter in the look-ahead modulator's optimization function is not (reasonably) possible.

Instead of calculating the cost of an output symbol on the basis of the actual prediction filter, it is possible to calculate the cost of an output symbol on the basis of a generic prediction filter. This generic prediction filter should be such that it, on average, improves the predictability of the bitstream, independent of the actual input signal to the modulator. Thus, the generic prediction filter should bias the selection procedure of the look-ahead modulator to prefer high frequency repetitive patterns over random high frequency patterns. The result will be that more strong high frequency tones will be produced, which can be easily predicted by the actual DST algorithm, thus improving the compression gain. Note that care should be taken to not generate strong tones at a too low frequency, since this could cause issues during playback if the reconstruction filter is not providing enough rejection of the tones. Furthermore, there is a large probability that the bitstream will not pass the SA-CD compliance tests defined in Annex D. If the tones are at a high enough frequency there will be no issue, since the power in the tones

will be similar to what can be generated under normal circumstances.

The operation of the proposed prediction filter, that will be used to construct a cost function, is as follows. On the basis of the most recent P feedback symbols that were added to the look-ahead path, a prediction is made for the next feedback value, similar to the approach followed in the DST algorithm. The P feedback symbols are all either '+1' or '-1', but the predicted value will be a fractional number. If the value is positive, the predicted feedback symbol is a '+1'. If the value is negative, the predicted feedback value is a '-1'. If the output of the prediction filter is zero there is no preference for a symbol. Thus, the larger the predicted value, the larger the preference for the predicted feedback value. Since only the sign of the prediction is important, optimizing for a predicted value with a magnitude of exactly unity will not be optimal as it will unnecessarily constrain the algorithm and reject proper solutions.

By combining the output of the prediction filter with the actual feedback symbol a cost value can be calculated. More specifically, if the actual feedback symbol is a '+1' and the prediction filter output is positive, the actual feedback value matches with the prediction, and no cost should be resulting. If the sign of the actual feedback value and of the prediction do not agree, a cost should be resulting. Since the magnitude of the prediction is an indication of the preference for a symbol, it can be used (with a proportionality factor) to calculate the prediction cost value for the actual selected feedback value. Instead of only adding a cost when the prediction and the actual feedback are not in agreement it is also possible to add a negative cost when the two are in agreement. Also in this case the magnitude of the predicted value can be used to derive the cost, since larger values indicate a stronger preference.

Since the DST algorithm uses a FIR filter to predict the future data, the prediction filter should also be FIR. However, since the objective is to generate bit patterns that are easy to predict, the prediction filter can be much shorter than the maximum length of 128 from the DST encoder. A short prediction filter has the additional benefit of causing only a minimal impact on the computational complexity of the look-ahead SDM.

10.3.2 Combining the cost functions

The normal cost function that defines the noise-shaping characteristic is independent of the prediction filter, and vice versa. Therefore, both cost functions are evaluated independently, i.e. in parallel, and the output consists of two values. The final selection function requires a single

cost value per path in order to decide which paths continue. Thus, the two cost values need to be combined to a single cost value by a third cost function. The combination should be such that under all circumstances the stability of the noise-shaping loop is the first priority, i.e. because of the improved predictability of the bitstream the system should not break. Furthermore, the combination should be such that there is only a minor and predictable, preferably constant, impact on the signal encoding quality. More specifically, the amount of quantization noise or distortion components should not vary wildly over time because of the varying predictability of the output. Thus, the prediction cost function should not have any other perceivable impact on the quality of the output than a minor decrease of the SNR at most. As a result, if desired, this reduction of the SNR could be counteracted by redesigning the loop filter.

A consequence of the above, in combination with the fact that the number of possibilities to encode a signal reduces with the amplitude of the signal, is that the impact of the prediction cost function on the final cost that is passed to the selection cost function should be inversely proportional with the signal amplitude. Thus, for large signals the prediction cost function should not contribute significantly to the final cost, but for low amplitude signals the prediction is allowed to have a significant impact on the output selection. Although not trivial to see, this functionality can be achieved by simply summing the two cost values, as depicted in fig. 10.5.

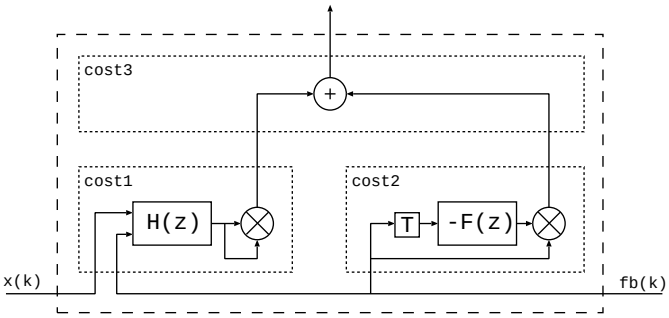


Figure 10.5: Combination of the main cost function (cost1) and the prediction cost function (cost2) by a third cost function (cost3).

The reason is the following. The input to the prediction filter consists of a fast toggling series of '+1' and '-1' values that describe the input signal in combination with the quantization noise. This signal has a constant

power. By applying a very steep low-pass filter to this signal the input signal can be obtained, and a variation of the filter output power as function of the signal amplitude can be seen. However, since the prediction filter is more or less an all-pass filter with some small notches at high frequencies (see next subsection), the output power of the prediction filter will hardly be affected by the signal amplitude. However, the output power of the normal noise-shaping filter will vary approximately proportionally with the signal amplitude. Since the output of the noise-shaping filter is subsequently squared to obtain the cost function output, the absolute value of the noise-shaping cost function will increase fast with the signal amplitude. Thus, if a properly scaled version of the prediction filter output is added to the squared output of the noise-shaping filter, the ratio between the two contributions will vary as a function of the signal amplitude, and the desired effect will be realized. Note that the multiplication of the negated output of the prediction filter with the feedback value in fig. 10.5 results in the addition of a cost value if the sign of the predicted value and the sign of the feedback value do not match, and a subtraction of the cost value if they match.

10.3.3 Spectral shaping

The addition of a second cost function, in parallel to the default cost function that evaluates the signal quality of the selected feedback values, will have an impact on the noise shaping that is realized by the modulator. In the normal look-ahead situation, i.e. with only loop filter H and no predictor cost function, the shaping of the quantization noise is, in first order, proportional to $\frac{1}{H}$ (sec. 5.6). This shaping is obtained since the optimization process attempts to minimize the sum of the squared filter output.

If we consider the situation of only the prediction filter F, a shaping of the quantization noise that is approximately proportional to $\frac{1}{1-z^{-1}F}$ is obtained. In this case the optimization process attempts to minimize the sum of the prediction error. Recall that the prediction error is negative (positive) when the sign of the prediction does (not) match the sign of the feedback symbol and that the magnitude of the prediction error is proportional with the magnitude of the output of the prediction filter. Since the sum of the prediction errors is minimized, and not the sum of the squares, the optimization process will attempt to find the bit sequence that results in the most negative prediction cost. The result of this optimization is different from the result that is obtained when the sum of the squares is minimized, i.e. in the sum of squares case the pre-

dicted signal matches as closely as possible the original signal whereas in this case the amplitude of the predicted signal is maximized. Therefore, the shaping of the quantization noise is not exactly proportional to $\frac{1}{1-z^{-1}F}$, but approximates this to a large extent, as will be demonstrated below.

The parallel combination of the two cost functions, realized by summing their costs, results in a shaping that is more or less proportional to $\frac{1}{H+\alpha(1-z^{-1}F)}$, where α is approximately inversely proportional to the input signal amplitude. The combination of the two filters does not exactly result in the parallel combination of the two since the output of the loop filter is weighed in a squared fashion, while the output of the prediction filter is weighed linearly in order to obtain the signal amplitude dependency. Furthermore, if the solution preferred by the prediction filter can not fulfill the noise-shaping criteria, the output of the noise-shaping filter will grow rapidly in the next clock cycles, and the solution will be rejected in favor of a solution that does fulfill the noise-shaping requirements. Thus, the shaping due to the prediction filter will only be realized if this is reasonably possible.

In order to obtain strong high frequency tones in the modulator's output spectrum, the prediction cost function should have a low output value for the frequencies for which tones are desired. This will result in a preference (low cost) for the bit sequences that have most of their energy at these specific frequencies over bit sequences that have no tones. In order to obtain a low cost value from the prediction cost function, the prediction filter should contain notches, i.e. the transfer of the prediction filter should be the inverse of the desired noise-shaping shape. If the notches are very narrow, clear well defined spectral tones will be resulting and a strong improvement of the compression gain can be expected in this case. However, in order to have narrow notches many filter coefficients are required. Furthermore, only few bit patterns will result in tones that match the frequencies of the notches if they are very narrow, and as a result only for a limited number of input signals the strong tones will be generated, since also the normal noise-shaping requirements need to be fulfilled. If the notches are slightly wider, there are more possibilities to generate bit patterns that describe the input signal and have high frequency tones. Overall this will result in a better predictability of the bitstream. If the notches become too wide the preference for specific frequencies will disappear and no improvement of the predictability will be realized. Thus, instead of realizing very narrow notches it is better to have slightly wider notches, but not too wide. Experimentally it has been determined that with a FIR prediction filter with eight coefficients a significant improvement of the predictability can be achieved, without

introducing a significant additional computational load.

An approximation of the spectral shaping of the residual error, obtained with an eight taps prediction filter that delivers good compression results (sec. 10.5.1), is plotted in fig. 10.6. Note that approximately the inverse transfer will be realized when the filter is inserted in the look-ahead optimization loop.

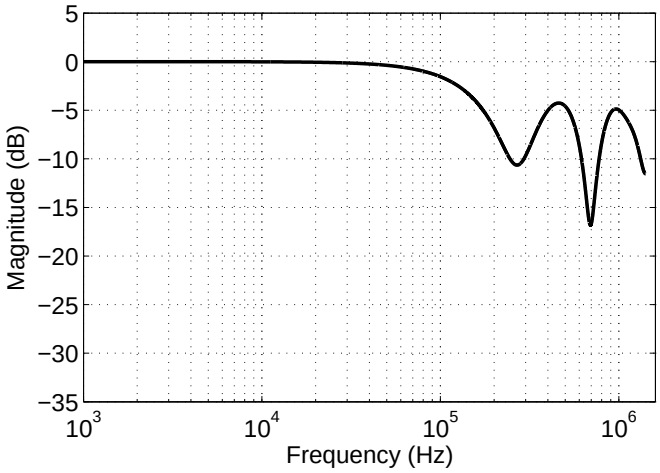


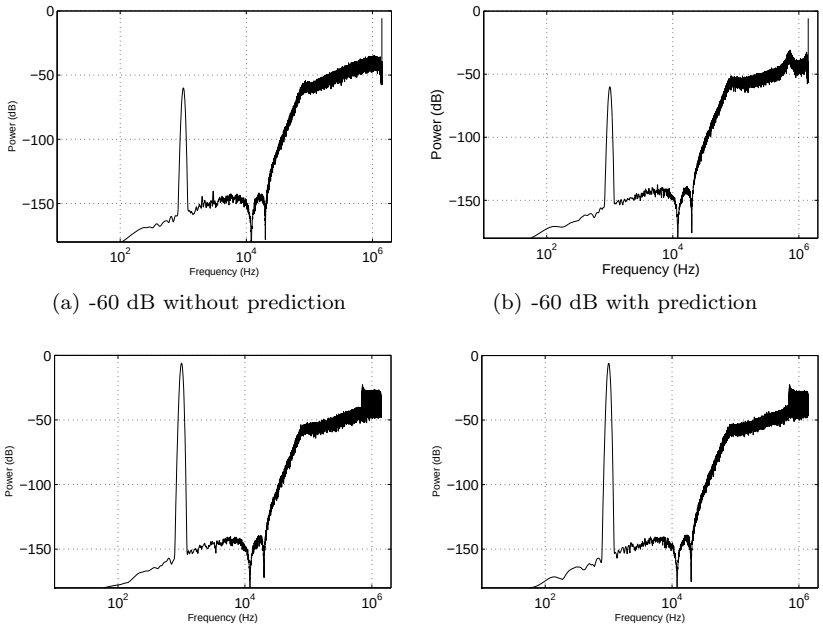
Figure 10.6: Spectral shape of the residual error resulting from the eight taps prediction filter.

With eight coefficients available for the prediction filter, only a limited number of possibilities for tuning the transfer, most importantly the bandwidth and the quality factor of the desired notches, is possible. Experimentally the following filter coefficients were determined:

$$b_1..b_8 = \{66, -91, -71, 42, -214, -255, -107, -2\}/255 \qquad (10.1)$$

The asymmetrical filter realizes a shallow notch around 250 kHz, a deep notch around 700 kHz, and another notch at half the sampling rate. The notch at half the sampling rate causes a preference for high frequency idle tones, as typically generated by an SDM. The other two notches stimulate the presence of tones in the low and middle part of the high frequency region. From these two notches the one around 700 kHz is the most effective because it is deeper, and its frequency is at Fs/4, which can be easily generated while at the same time satisfying the noise-shaping criteria.

In fig. 10.7 the effect of the prediction filter on the spectrum is demonstrated for a -60 dB input signal (without prediction filter in fig. (a), with in fig. (b)) and for a -6 dB signal in figs. (c) and (d). Clearly, in the case of the -60 dB signal the prediction filter changes the noise-shaping properties and causes a peak in the noise shaping at 700 kHz. In the case of the -6 dB signal hardly any differences can be detected between the two spectra.



(a) -60 dB without prediction

(b) -60 dB with prediction

(c) -6 dB without prediction

(d) -6 dB with prediction

Figure 10.7: The effect of the prediction filter on the spectrum is clearly visible for the -60 dB input signal. No significant difference is present for the -6 dB signal.

10.4 Algorithm

A look-ahead modulator implementation that performs a dual optimization can be based on any of the previously described look-ahead algorithms. Since the Pruned Tree sigma-delta modulation algorithm is the computationally most efficient one from all the investigated approaches, and since it delivers a very good signal conversion performance

with excellent stability, the new algorithm that performs a dual optimization will be based on the original Pruned Tree sigma-delta modulation algorithm. The new algorithm is called Pruned Tree sigma-delta modulation for SA-CD, to indicate it is the original Pruned Tree sigma-delta modulation algorithm with an addition for Super Audio CD. The algorithm is published in [30,34,49], although the important details of the algorithm are not disclosed in the paper.

The main actions that are performed in the Pruned Tree sigma-delta modulation algorithm for SA-CD are equal to those of the original Pruned Tree sigma-delta modulation algorithm, see sec. 9.2. First there is the initialization phase that sets the proper initial conditions for the algorithm. Then there is the operation phase for the actual conversion. The only difference between the two algorithms is a detail of the cost function, i.e. the cost function is more complex since it takes the predictability of the solution under investigation into account. Since there are no other differences please refer to sec. 9.2 for a detailed description of all the steps of the algorithm.

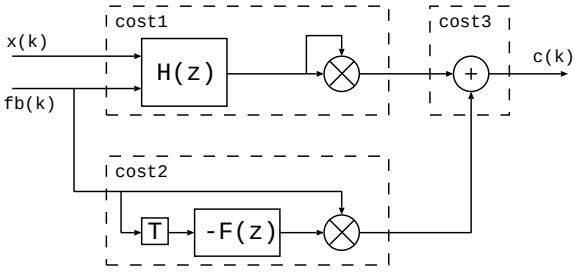


Figure 10.8: Look-ahead filter with loop filter H (cost1), prediction filter F (cost2), and final cost value c(k).

In fig. 10.8 an alternative graphical representation is shown of the look-ahead filter, consisting of the main loop filter H with the primary cost function (cost1), the prediction filter F with the prediction cost function (cost2), and the combining cost function (cost3), resulting in the final cost value c(k).

A comparison to fig. 10.9 that shows how dither can be applied to a look-ahead filter (reproduced from fig. 7.29 with minor modifications), reveals that the addition of the prediction cost value to the main cost value is performed in the same way as the addition of a dither value to the main cost function. In the case of the structure with the prediction cost function, the dither is not a random signal but a deterministic signal

that is derived from the SDM output (feedback) signal.

If a dither signal is completely random, it reduces the correlation between the quantization errors, resulting in a white quantization error spectrum if there is no noise shaping. The effect of the dither is independent of the frequency content of the quantization errors. The effect of the prediction cost function is similar to the effect of normal dither, with the difference that the effect of the "dither" is now a function of the frequency content of the combined input signal and the quantization error. Because of how the prediction cost value is calculated, it causes an increase in the correlation for certain specific high frequencies, but for the low frequency content it is reducing the correlation of the quantization errors. This can be understood from the following reasoning.

If it is assumed that the high frequency quantization noise is approximating actual random noise and is not correlated with the input signal, the prediction cost value does not depend on the (low frequency) input signal, since most of the signal power that is used to derive the prediction cost value is describing the high frequency frequency region. In reality, the high frequency quantization noise is correlated with the low frequency input signal, but the relationship is very complex and can not be considered as a simple mapping. As a result, in a first-order approximation, the high frequency quantization noise can be considered uncorrelated with the input signal. Thus, if the output bitstream, consisting of the sum of the (small) input signal and the (large) quantization errors, is used to derive a dither signal via the prediction filter, the dither will be virtually uncorrelated with the input signal. As a result, the prediction cost function, that is designed to increase the correlation of the high frequency content, will dither and de-correlate the low frequency quantization errors.

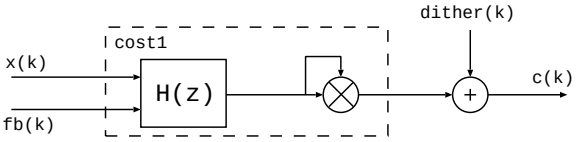


Figure 10.9: Preferred solution for adding dither to a look-ahead filter (from fig. 7.29 with minor modifications).

Although there are some results in literature that show that signal dependent dither can be beneficial in the case of a 1-bit SDM [43–45], according to the traditional theory of dithered quantization the dither signal and the signal to be quantized should be statistically independent

in order to avoid spectral modulation [19, 42, 63, 64]. In the case of a 1-bit SDM there is, by definition, spectral (noise) modulation since the total output power is constant while the input power is not. If only the signal band is considered, i.e. the band between 0 and 20 kHz, there is still modulation of the quantization noise as a function of the signal power, as illustrated in the previous chapters. However, by applying the signal dependent dither that increases the predictability of the output signal, it is possible to reduce at the same time the level of in-band noise modulation and to reduce the non-linearities (harmonic distortion), as will be demonstrated in the next section.

10.5 Functional performance

In this section the lossless data compression performance, as well as the classical and SDM specific performance of the Pruned Tree SDM for SA-CD is evaluated.

10.5.1 Lossless data compression

The effectiveness of the dual cost function approach with respect to improving the compression gain of the output bitstream is investigated by comparing the compression gain to that obtained with a normal SDM and a normal Pruned Tree SDM. For practical reasons, the compression gain is calculated by compressing the binary output stream with the (fast) ZIP algorithm, known from the popular ".ZIP" file format [48], instead of with the computationally very intensive DST algorithm. Alternatively, it would be possible to calculate the minimum number of required bits to store the modulator's output bitstream by calculating the entropy of the signal. Since this approach requires more work, and does not bring additional insight, the ZIP algorithm is used instead. These results are therefore a first order indication of the actual compression gain that will be achieved by the DST compression algorithm, although they are slightly optimistic. The reason for this difference in effectiveness is that the DST algorithm is designed to realize good compression gains for 1-bit encoded data with a minimal requirement on the amount of hardware resources for decoding. Encoding on the other hand, is very computational intensive because of the optimization of the prediction filter. The ZIP algorithm can make use of extensive hardware resources for both encoding and decoding and realizes, typically, a higher compression gain in far fewer computations.

In the experiment a comparison of the compression gain is made for a 1 kHz sinusoid with signal levels ranging from -110 dB to -5 dB. The two Pruned Tree Sigma-Delta Modulators are configured with eight parallel paths. The loop filter of all the three modulators is equal to loop-filter configuration SDM2 (app. B). The results are depicted in fig. 10.10.

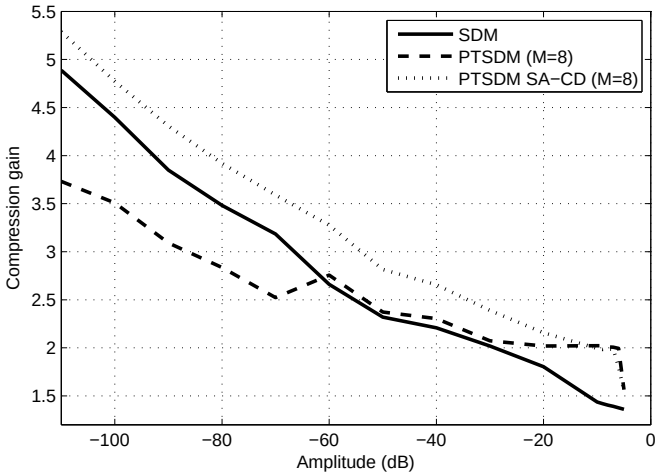


Figure 10.10: Compression gain of a normal SDM, a Pruned Tree SDM (M=8), and a Pruned Tree SDM for SA-CD (M=8) as a function of the signal level. The loop filter is SDM configuration SDM2.

A clear difference in the compression gain can be observed between the three different modulators. The Pruned Tree SDM for SA-CD with $M = 8$ realizes a compression gain that is approximately 0.4 higher than that of the SDM, independent of the signal level. This is an increase of 10-40%, depending on the signal level. The normal Pruned Tree SDM (M=8) realizes a compression gain that is much lower than that of the normal SDM for signal levels below -60 dB, most likely because of the significantly higher SNR the converter realizes in this region (see sec. 10.5.4). Between -60 dB and -30 dB the compression gain of the Pruned Tree SDM is about equal to that of the SDM, and only for signals larger than -30 dB the Pruned Tree SDM encoding results in a gain that is higher than that of the normal SDM. Thus, the modification of the Pruned Tree sigma-delta modulation algorithm to take the predictability of the bitstream into account has clearly a positive impact on the compression gain.

Instead of studying the compression gain as a function of the signal level, the compression gain is measured as a function of the number of parallel paths. Fig. 10.11 shows the outcome of this experiment for three different signal levels.

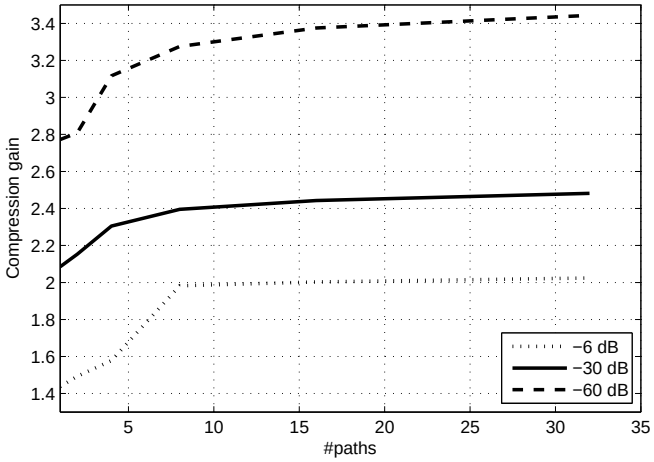


Figure 10.11: Compression gain of a Pruned Tree SDM for SA-CD as a function of the number of parallel paths for various signal levels. The loop filter is SDM configuration SDM2.

For all three signal levels the compression gain increases when the number of parallel paths is increased. The increase is the strongest for the first eight paths. If the number of paths is further increased the compression does increase, but the improvement per path becomes much less, especially for high signal levels. Thus, already with a few paths a large benefit from the optimization for predictability is obtained.

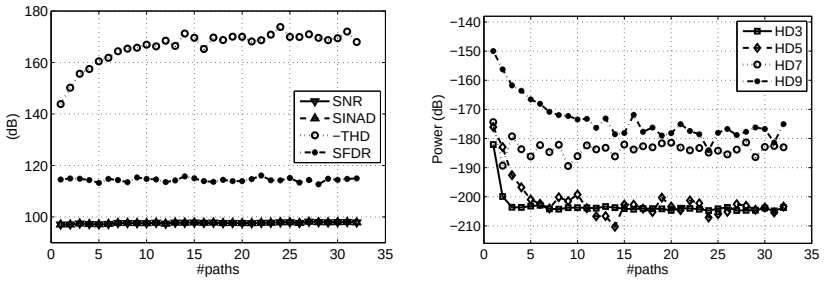
10.5.2 SNR, SINAD, THD and SFDR

The impact of the prediction cost function on the traditional signal quality indicators, i.e. the SNR, the SINAD, the THD, and the SFDR, is investigated as a function of the number of parallel paths. Similar to the approach followed in the previous two chapters, the impact is measured for SDM configuration SDM1 and SDM2 (see app. B). For all FFT calculations a length of 1 million samples is used. To get reliable read-

ings of the harmonic distortion components 128 coherent averages are performed in combination with 4 power averages.

SDM1

The SNR, the SINAD, the THD, and the SFDR that are obtained for a 1 kHz sine wave with an amplitude of -6 dB for SDM configuration SDM1, as a function of the number of parallel paths, are depicted in fig. 10.12(a). The buildup of the THD can be found in fig. 10.12(b) that shows the power of the first four odd signal harmonics.



(a) SNR, SINAD, THD, SFDR.

(b) HD3, HD5, HD7, HD9.

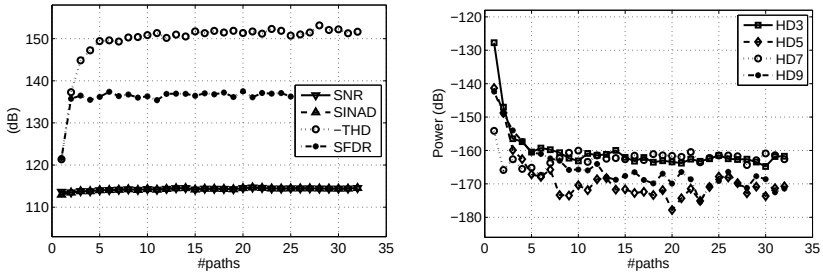
Figure 10.12: SNR, SINAD, THD, and SFDR performance for configuration SDM1 as a function of the number of parallel paths for a 1 kHz sine wave with a power of -6 dB (a) and the power in HD3, HD5, HD7, and HD9 (b) for a history length of $L = 1024$ samples.

A comparison of the results to those obtained for the normal Pruned Tree SDM (fig. 9.3) reveals that the SNR, the SINAD, and the SFDR curves are very similar but not equal. More specifically, the SNR and the SINAD realized by the Pruned Tree SDM without the prediction filter are approximately 0.5 dB higher (the SFDR 1 dB) than what is achieved by the Pruned Tree SDM with the prediction filter, independent of the number of parallel paths. Thus, the prediction filter adds a small amount of noise in the baseband region. However, the THD improves much faster than with the normal Pruned Tree SDM. For example, with eight paths the THD is already at -165 dB, a level that the normal Pruned Tree SDM can only reach with 32 paths. This difference can also be clearly recognized when the power in the harmonic components is compared, i.e. already in the case of $M = 1$ the level of all the harmonics, except for the ninth, is much lower for the Pruned Tree SDM for SA-CD. Note that the SFDR is limited by the quantization noise floor, independent

of the number of parallel paths, and that the harmonic distortion is at a much lower level.

SDM2

In fig. 10.13(a) the SNR, the SINAD, the THD, and the SFDR performance, obtained for SDM configuration SDM2 as a function of the number of parallel paths, is depicted. The power of the first four odd harmonics is shown in fig. 10.13(b).



(a) SNR, SINAD, THD, SFDR.

(b) HD3, HD5, HD7, HD9.

Figure 10.13: SNR, SINAD, THD, and SFDR performance for configuration SDM2 as a function of the number of parallel paths for a 1 kHz sine wave with a power of -6 dB (a) and the power in HD3, HD5, HD7, and HD9 (b) for a history length of $L = 1024$ samples.

For SDM configuration SDM2 a similar situation is observed as for SDM configuration SDM1. The SNR and the SINAD are again 0.5 dB lower than what is obtained with the normal Pruned Tree SDM, while the SFDR is approximately 1 dB lower. The THD improves much faster than with the normal Pruned Tree SDM and with only eight parallel paths the THD is already at a level of -150 dB, a level that can just be realized with 32 paths by the normal Pruned Tree SDM. Also for this SDM configuration the SFDR is already limited by the quantization noise floor with two parallel paths. A further increase of the number of parallel paths does result in a larger reduction of the harmonic distortion content, pushing them far below the quantization noise floor.

10.5.3 Converter stability

The stability of the Pruned Tree SDM for SA-CD is investigated and compared to the stability of the original Pruned Tree SDM. The stability is characterized by determining the maximum input amplitude that can be converted and by the maximum loop-filter corner frequency that can be used.

Maximum stable input amplitude

The maximum amplitude of a a 1 kHz sine wave that can be converted by the Pruned Tree SDM for SA-CD converter is measured, as a function of the number of parallel paths. The loop filter used is SDM configuration SDM2, i.e. a 100 kHz filter with two resonator sections (app. B). The results of this measurement are depicted in fig. 10.14 and compared to the results obtained for the normal Pruned Tree SDM (from fig. 9.6(a)).

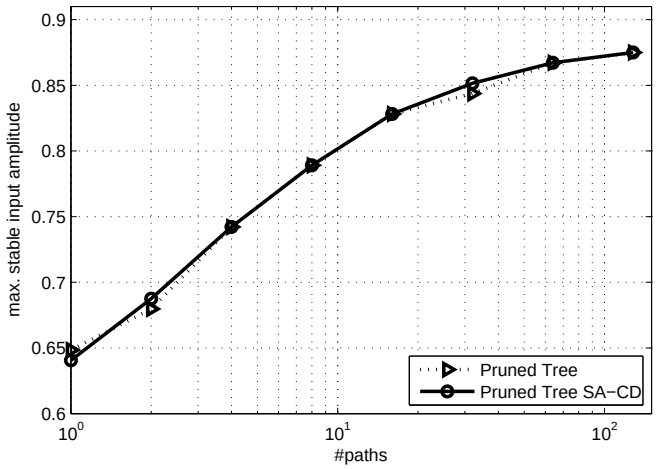


Figure 10.14: Maximum stable input amplitude as a function of the number of parallel paths for a Pruned Tree SDM and a Pruned Tree SDM with prediction filter for configuration SDM2 for a 1 kHz sine wave.

The two curves are nearly identical for all measurement points, and no significant differences can be detected. Thus, as predicted, for large signal levels the prediction cost function has virtually no impact on

the final cost value and the stability of the converter is not degraded compared to a converter without prediction filter.

Maximum loop-filter corner frequency

For a 1 kHz sine wave with a level of -6 dB the maximum corner frequency that results in stable operation, as a function of the number of parallel paths, is investigated. The result of this experiment is compared in fig. 10.15 to the result obtained for the normal Pruned Tree SDM (from fig. 9.7(a)).

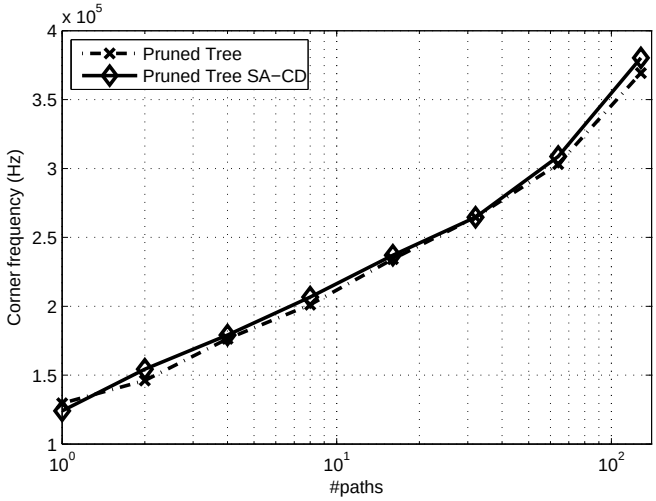
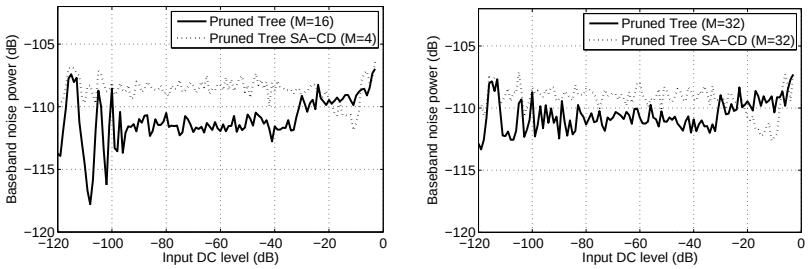


Figure 10.15: Maximum loop-filter corner frequency that results in stable operation as a function of the number of parallel paths for a Pruned Tree SDM and a Pruned Tree SDM with prediction filter for a -6 dB 1 kHz sine wave with filter configuration SDM2.

No significant difference in the maximum corner frequency that still results in stable operation can be detected between the Pruned Tree SDM with and without prediction filter. This again confirms that the addition of the prediction filter does not affect the operation of the converter when it is pushed to its extreme.

10.5.4 Noise modulation

From the experiments described in sec. 10.5.2 it is known that for a -6 dB input signal the power of the baseband noise is 0.5 dB higher for the Pruned Tree SDM with prediction filter than for the normal Pruned Tree SDM if the same number of paths is used. Since the prediction filter is mainly influencing the output selection for low level input signals, it is expected that the prediction filter has an impact on the amount of in-band noise for low level signals as well. This effect, in combination with the traditional problem of noise modulation is investigated as a function of the DC input level.



(a) $M = 16$ vs. $M = 4$ SA-CD

(b) $M = 32$ vs. $M = 32$ SA-CD

Figure 10.16: In-band noise as a function of the DC input level. Comparison of a Pruned Tree SDM with $M = 16$ ($M = 32$) to a Pruned Tree SDM with a prediction filter with $M = 4$ ($M = 32$) in fig. (a) (b) for a logarithmic input level selection. The loop filter is SDM configuration SDM1.

In fig. 10.16 for loop-filter configuration SDM1 (app. B) the amount of baseband noise power for input levels between -120 dB and -3 dB is plotted for a normal Pruned Tree SDM and for a Pruned Tree SDM with prediction filter. In fig. 10.16(a) the normal Pruned Tree SDM is configured with $M = 16$ and the Pruned Tree SDM for SA-CD with $M = 4$. For input levels up to -30 dB the modulator with the prediction filter has around 7 dB more baseband noise power. From -25 dB till -11 dB the amount of in-band noise of the Pruned Tree SDM with prediction filter reduces and the noise power becomes lower than that of the normal Pruned Tree SDM at an input level of -20 dB. Finally, the noise level increases again to go above that of the normal Pruned Tree SDM at an input level of around -8 dB.

If both the modulators are configured with 32 parallel paths instead,

an interesting change in the noise levels is observed, see fig. 10.16(b). In the case of the modulator without prediction filter (M=32) the noise level increases by approximately 1.5 dB for low level inputs, compared to the situation of no prediction filter and M=16. For high level inputs the noise level reduces by nearly 1 dB. Note that these differences are almost too small to be noticeable in the figures. In the case of the modulator with prediction filter a different behavior is observed. More specifically, for all input levels the noise level reduces a fraction of a decibel. Only for inputs between -20 dB and -10 dB the difference is large enough to cause a noticeable difference in the SNR. Thus, while the in-band noise level of the Pruned Tree SDM for SA-CD is almost insensitive to the number of parallel paths, the normal Pruned Tree SDM shows a large variation of the in-band noise. If more paths are used the amount of noise modulation of the normal Pruned Tree SDM reduces and the noise behavior of the two modulators becomes almost equal, except for large input signals.

In the curves describing the in-band noise of the Pruned Tree SDM for SA-CD three different regions can be recognized. For low input amplitudes the prediction filter has a large impact on the selection of the output symbols, and the dither signal adds some noise to the output, resulting in a noise level that is higher and more constant than that of the normal Pruned Tree SDM. Then there is a transition region where the amount of in-band noise reduces. In this region the contribution of the normal cost function starts to become dominant, but the prediction filter still influences the output symbol selection. The combination of the two cost functions is such that for this specific range of input levels a limited number of high frequency tones are generated that cause the amount of baseband quantization noise to reduce. However, if the input signal level is increased slightly more the normal noise-shaping cost function takes over and dictates the output symbol selection, with a minor noise penalty from the prediction filter compared to the normal Pruned Tree SDM.

Instead of measuring the in-band noise power for DC signals it is also insightful to measure the SNR as a function of the signal level. Ideally a converter should generate a constant amount of in-band noise, independent of the signal level, and the result of the measurement should be a straight line. In the case of the Pruned Tree SDM for SA-CD the result of the measurement is very close to this ideal situation, independent of the number of parallel paths, as demonstrated in fig. 10.17 for SDM configuration SDM2 (app. B). Only in the case of $M = 32$ a small deviation from the ideal curve, i.e. an approximately 1.5 dB higher SNR, is present for inputs around -10 dB. This deviation is too small to be noticeable

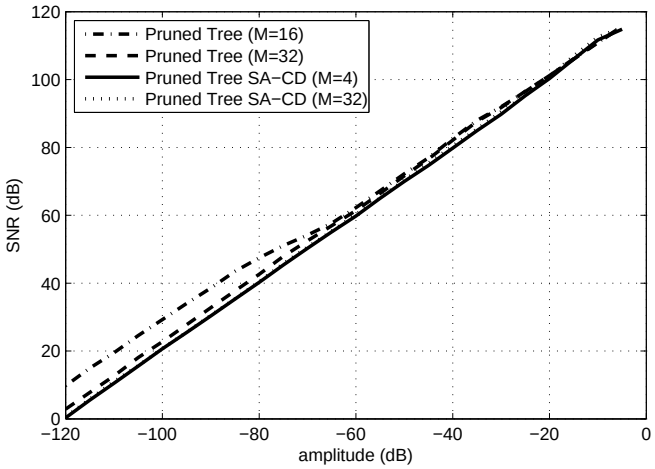


Figure 10.17: Comparison of the obtained SNR for a Pruned Tree SDM with $M = 16$ and with $M = 32$ to that of a Pruned Tree SDM with a prediction filter with $M = 4$ and $M = 32$ (curves overlap). The loop filter is SDM configuration SDM2.

in the plot, and the curves for the Pruned Tree SDM for SA-CD with $M = 4$ and $M = 32$ are virtually indistinguishable. The normal Pruned Tree SDM shows a large deviation from the ideal SNR curve, especially when 16 parallel paths are used. When 32 parallel paths are used the converter realizes a lower SNR for small input signals, but approximates the ideal transfer better and becomes more linear as demonstrated in the previous chapter. If the number of parallel paths is increased further it is expected that the in-band noise level for low level inputs will also increase further to end up at approximately the same level as of the converter with the prediction filter. Thus, although the Pruned Tree SDM for SA-CD realizes a slightly lower SNR than the normal Pruned Tree SDM it does exhibit less noise modulation, independent of the number of parallel paths.

10.5.5 Summary

The Pruned Tree SDM for SA-CD was specifically designed with the objective to increase the lossless compression gain of the modulator's output bitstream. Experiments on sinusoidal signals show that indeed

the Pruned Tree SDM for SA-CD realizes bitstreams that result in a significantly higher lossless compression gain than obtained with a normal Pruned Tree SDM or a traditional SDM. Depending on the signal level an increase between 10 and 40% can be obtained. If more parallel paths are used the compression gain increases, with the biggest improvement realized when the number of paths is increased from one to eight. In the case of low amplitude signals the compression gain can be increased more by increasing the number of paths further, although here the biggest improvement is already obtained for four parallel paths.

The prediction filter, responsible for the increase in compression gain, can also be considered as a signal dependent dither. According to the traditional theory of dithering this should result in non-linearities. However, the THD measurements reveal that the Pruned Tree SDM for SA-CD is more linear than the normal Pruned Tree SDM. With only a few parallel paths the harmonic distortion components are suppressed by more than 20 dB, an improvement that can barely be realized with a normal Pruned Tree SDM with 32 paths. The addition of the prediction filter also brings some small disadvantages. More specifically, compared to the normal Pruned Tree SDM the SNR and the SINAD reduce by 0.5 dB for a -6 dB input signal. The impact on the SFDR is approximately 1 dB.

Although the prediction filter always attempts to bias the signal selection, it is only effective in this for low amplitude signals. The higher the signal amplitude, the smaller the impact of the prediction filter on the output selection. As a result, there is no negative impact on the stability of the converter, and the same maximum signal amplitude can be converted as with the normal Pruned Tree SDM. Compared to the normal Pruned Tree SDM also no difference in the maximum loop-filter corner frequency that can be used is found, again confirming that the biasing of the output selection is mainly active for low amplitude signals.

The amount of noise modulation realized by the Pruned Tree SDM for SA-CD is less than that of the normal Pruned Tree SDM. Because of the prediction filter the level of the in-band noise for low amplitude signals is virtually independent of the number of paths, whereas in the case of the normal Pruned Tree SDM the noise level varies with the number of paths. This result is confirmed by measuring the SNR for sinusoids as a function of the signal level. In the case of the Pruned Tree SDM for SA-CD a virtually perfect SNR vs. amplitude transfer is obtained, while the normal Pruned Tree SDM can show a large deviation from the perfect line, depending on the number of parallel paths.

10.6 Implementation aspects

Since the Pruned Tree SDM for SA-CD is based on the normal Pruned Tree SDM, the same strategy for an efficient implementation should be followed as described in sec. 9.5, with the addition of the prediction cost function. Several aspects related to an efficient evaluation of the prediction cost function are outlined below. Note that incorporating the prediction cost function requires the evaluation of the output of the prediction filter, combining this output with the trial feedback value, and adding the resulting prediction cost to the output of the main cost function.

A straightforward evaluation of the prediction filter would perform a multiply and accumulate operation for each of the prediction filter coefficients. In a more optimized implementation these operations could be simplified to an addition or subtraction operation per filter coefficient, depending on the bit value. However, in a (software) implementation the prediction filter output evaluation can be performed much more efficiently by using a lookup table. More specifically, since the input data to the prediction filter with N coefficients is a string of N 1-bit values, these N 1-bit values can be combined to an N-bit word. Subsequently this N-bit word can be used to access a table that contains the (pre-computed) 2^N possible output values of the prediction filter. As long as the value of N is reasonably small, e.g. 8 to 16 bits, the memory overhead of this approach is small, and a very efficient evaluation of the prediction filter output is realized.

Alternatively, if the number of filter coefficients is too large, i.e. if the lookup table becomes prohibitively big, the prediction filter output can be calculated by combining the output of multiple smaller tables with less index bits and adding the results. For example, the response of a filter with 16 coefficients can be obtained by combining the output from two lookup tables with an 8-bit index. Such an approach reduces the number of pre-computed values from 2^{16} to 2^9 at the cost of a single addition and two table lookups instead of one.

Because the input to the prediction filter only consists of the N most recently added feedback symbols, the output of the prediction filter is independent of the trial feedback symbol under evaluation. Thus, the prediction filter output only needs to be calculated once per path and can be reused for both trial feedback symbol evaluations. However, the final prediction cost value depends on the feedback symbol and is obtained by multiplying the output of the filter with the feedback symbol and inverting the sign. Finally, the prediction cost value is added to the

main cost value to obtain the total cost for the trial feedback value. In an optimized look-ahead filter implementation the cost value for the '+1' and '-1' symbol will be calculated in parallel with minimal overhead (see sec. 7.6.2), and in this case the output of the prediction filter can simply be subtracted (added) from the main cost for the '+1' ('-1') feedback value.

10.7 Conclusions

In order for an SDM to be suitable for the generation of SA-CD content it needs to fulfill a number of requirements. All these requirements can be satisfied without any problem by all of the earlier described look-ahead modulators. However, the requirement of generating bitstreams that can be compressed well, i.e. have a high predictability, can not be fulfilled by any of these modulators. By adding a cost function to the look-ahead filter that takes the predictability of the output bitstream into account, and by properly combining this cost with the main cost that is an indicator of the encoding quality, it is possible to generate high quality encodings that can be compressed well. The resulting algorithm is the Pruned Tree sigma-delta modulation algorithm for SA-CD.

In the algorithm a simple generic prediction filter is used instead of the complex prediction filter that can be used by the DST algorithm. On the basis of the eight most recent feedback symbols a prediction is made of the current feedback symbol, and a cost is calculated that reflects the quality of the match. This cost is added to the main cost, effectively resulting in a signal dependent dither. The effect of this approach is that the prediction filter stimulates the creation of high frequency tones, such that the predictability of the bitstream increases, but that no in-band distortion is resulting. It has been demonstrated that the approach is effective and that, depending on the signal amplitude and the number of parallel paths, an improvement in the compression gain between 10 and 40% can be achieved in comparison to a normal SDM and to a Pruned Tree SDM. The improvement is the largest for low amplitude signals since here there is a lot of freedom in selecting bit patterns without violating the noise-shaping criteria, whereas for large signals the stability of the converter is reducing fast and the impact of the prediction cost on the output selection is smaller.

Besides improving the predictability of the output of the converter, the addition of the prediction filter also has a small impact on the signal conversion quality. More specifically, the SNR and the SINAD of the converter reduce by approximately 0.5 dB for large signals. The penalty

on the SFDR is 1 dB. However, the THD of the converter improves significantly, and with only eight parallel paths the same level of distortion can be obtained as with a Pruned Tree SDM with 32 paths. Since the prediction filter mainly influences the feedback symbol selection for small signals, the stability of the converter is not influenced, and the same performance as with a normal Pruned Tree SDM is obtained. Finally, the addition of the prediction filter has a positive impact on the noise modulation performance. Virtually independent of the number of paths, the amount of in-band noise is constant for signal levels between -120 dB and -20 dB. As a result there is no noticeable noise modulation present and a perfect SNR vs. signal level curve is obtained. Thus, with minimal computational overhead a modulator has been realized that is ultimately suitable for the generation of high quality 1-bit encoded audio signals for storage on Super Audio CD.

Chapter 11

Comparison of look-ahead SDM techniques

In this chapter all the look-ahead techniques that were discussed in detail in the previous chapters are compared. First, in sec. 11.1 an analysis is made of alternative look-ahead techniques published in literature to determine if these should be included in the comparison. The outcome of this comparison is that all the published results are either covered by this work or that the approaches can not be extended to high order Sigma-Delta Modulators, on which the focus of this work is. In sec. 11.2 the algorithms of the in this work discussed techniques are compared. Their functional performance is evaluated in sec. 11.3. Finally, conclusions are drawn in sec. 11.4.

11.1 Alternative look-ahead techniques

A literature search reveals that there are many publications [3–6, 18, 21–25, 60, 65] on look-ahead sigma-delta modulation for digital-to-digital conversion. However, nearly all of these algorithms are equal or very similar to the in this work already described algorithms.

For example, the moving-horizon optimal quantizer [18] is an alternative implementation of the full look-ahead algorithm. In [6] the full tree algorithm and the stack algorithm are described, which both implement a full look-ahead. In the same paper also the Fano algorithm is presented, which is an approximation to the full look-ahead sigma-delta modulation algorithm, and it will therefore result in a quality that is in the best case equal to that of the full look-ahead algorithm.

In [25] a look-ahead algorithm is presented that is equal to the Pruned Tree sigma-delta modulation algorithm, but without the check for convergence. As a result, an extremely long history length is required in order to guarantee stability of the algorithm, and no improvement compared to the Pruned Tree sigma-delta modulation algorithm of ch. 9 can be realized.

In [23, 24] an alternative sigma-delta modulation technique is demonstrated that derives the quantizer decision on the total energy of the loop-filter states instead of on the weighted sum of the states. This technique results, in combination with traditional full look-ahead, in a larger stability of the converter compared to standard sigma-delta modulation. Although in this approach the sigma-delta modulation technique is not standard, the look-ahead technique is the basic full look-ahead approach, and the algorithm will not be included in further comparisons.

However, there are also publications that present look-ahead algorithms that are different from the ones already discussed. In [60] a look-ahead algorithm is presented that attempts to estimate the impact of the quantizer decision on the future stability of the loop filter. The feasibility of the approach is demonstrated for loop-filter orders up to three, and an improved stability is claimed. However, an actual quality comparison with any of the in this work described solutions is not possible since it is not demonstrated or clear how the approach can be extended to higher order filters.

In [21, 22] a step-back algorithm is presented. Although this algorithm does not perform actual look-ahead, it is related to the class of look-ahead algorithms since it can change the output symbol on the basis of its impact on the future. More specifically, this algorithm is normally performing standard sigma-delta modulation, but when instability is detected an alternative encoding for the last series of bits is performed. This is realized by stepping back in time a number of clock cycles, changing the selected output symbol, and continuing the encoding from there. Obviously, this approach does not guarantee that the instability will be avoided, and often multiple step-back operations need to be performed in order to find a solution that is stable, as demonstrated in the original publication. Furthermore, since the algorithm performs normal sigma-delta modulation as long as no instability is detected, the signal encoding quality of the algorithm is equal to that of a normal SDM, except that a larger stability can be realized. Since no improvement in the SFDR or THD can be realized, in addition to the disadvantage of the non-constant throughput rate of the algorithm, the algorithm will not be studied further.

From the above it is clear that, to the best knowledge of the author, there are no interesting alternative look-ahead techniques that can be compared to the ones presented in this work. Therefore, in the next sections a comparison will only be made between the in this thesis described solutions.

11.2 Algorithm comparison

In ch. 5 the general look-ahead concept has been investigated, and the full look-ahead sigma-delta modulation algorithm has been presented. On the basis of this algorithm the Trellis sigma-delta modulation algorithm (ch. 7), as well as the even more efficient look-ahead algorithms, i.e. the Efficient Trellis sigma-delta modulation algorithm (ch. 8), the Pruned Tree sigma-delta modulation algorithm (ch. 9), and the Pruned Tree sigma-delta modulation algorithm for SA-CD (ch. 10), have been derived. As a result of the pruning that is performed in these algorithms they all realize an improvement over the full look-ahead algorithm, both from a computational point of view and also from a signal conversion quality point of view, as demonstrated in sec. 11.3.

From an algorithmic point of view the difference between the full look-ahead sigma-delta modulation algorithm and the pruned look-ahead techniques is very clear, i.e. there is no pruning. More specifically, in the full look-ahead approach the path cost of every possible solution in the look-ahead interval is evaluated in order to decide on the next output symbol. If there is no constraint on the amount of computational resources this approach will result in the highest conversion quality since every possible solution is evaluated. In practice there will be only limited computational resources available, and as a result the amount of look-ahead that can be realized is small. The algorithms that perform pruning will not evaluate every possible solution and can realize a larger amount of look-ahead with the same amount of resources. However, because no exhaustive search is performed it is possible that relevant solutions are not investigated, resulting in a lower conversion quality than what is possible. If the metric that is used to perform the pruning is of a good quality this situation will not occur frequently and overall an improvement in the conversion quality will be obtained.

A comparison of the algorithmic steps of the four pruning look-ahead solutions immediately reveals that the algorithms are very similar. In fact, the main difference between the algorithms is the selection procedure that determines which solutions are continuing. In other words,

the selection cost function differs[1]. All the other steps of the algorithms are comparable and differ only in their details, because of the different selection cost functions. With these details removed, the operations performed by all the studied look-ahead algorithms can be described as:

1. extend all the paths with a '0' and a '1' bit and calculate the cost values;
2. calculate the accumulated path cost values;
3. select the paths that continue and update the look-ahead filter states;
4. adjust the path metric values;
5. determine the output symbol;
6. invalidate the paths that have not converged.

Although the impact of the selection cost function might seem minor, it can have a large impact on the computational load and the functional performance of a converter. For example, in the Trellis sigma-delta modulation algorithm, because of the selection strategy, the number of parallel solutions to investigate can not be selected arbitrarily but only as a power of two. A result of this is that in order to realize minimally more look-ahead, the computational load will double. The Efficient Trellis sigma-delta modulation algorithm and the Pruned Tree sigma-delta modulation based algorithms have a different selection criterion that makes it possible to select any number of parallel paths, and as a result their computational load is much more scalable.

Besides an impact on the number of paths, the selection criterion can also have an impact on the computational load per path. For example, although in the Efficient Trellis sigma-delta modulation algorithm the number of parallel paths can be selected without any constraint, the selection criterion does result in a computational load that scales more than linear with the number of paths. In the Pruned Tree sigma-delta modulation algorithm this problem has been tackled by relaxing the selection criterion in combination with a proper initialization of the system. As a result of this change in the cost function not only the computational load has decreased, but at the same time an improvement in

[1] In the case of the Pruned Tree sigma-delta modulation algorithm for SA-CD the selection cost function is equal to that of the normal Pruned Tree sigma-delta modulation algorithm, but the algorithms uses a different path cost function that includes a measure for the predictability of the bitstream.

the functional performance has been realized. Thus, the selection criterion can have an impact on the number of parallel paths that can be used, on the computational load per path, and also on the functional performance of the look-ahead algorithm (see sec. 11.3).

The computational load of the different look-ahead algorithms can be qualitatively compared by selecting the number of parallel paths M as a power of two, i.e. $M = 2^N$, such that the number of paths for each of the algorithms becomes equal. Under these assumptions the full look-ahead sigma-delta modulation algorithm has the lowest computational load because the output symbol can be found by simply selecting the cheapest path without any calculations or memory updates related to pruning. The Trellis sigma-delta modulation algorithm requires more computations since here the selection procedure consists of selecting M times between two paths, but also a memory update is required to keep track of the solutions under investigation. The Pruned Tree sigma-delta modulation algorithm is again slightly more expensive since here the selection procedure determines the M cheapest paths from the $2M$ available paths. This requires a sorting operation that is more expensive than comparing M times two values, but that can be made nearly linear with M (sec. 9.5). The Pruned Tree sigma-delta modulation algorithm for SA-CD is minimally more expensive than the normal Pruned Tree sigma-delta modulation algorithm since the cost function that evaluates the quality of the bitstream is more complex. Finally, the Efficient Trellis sigma-delta modulation algorithm is the most expensive, especially if M is large. This algorithm is more expensive than the other algorithms because it does not only determine the M cheapest paths, but it also has to compare the selected M paths to make certain that they are all unique in their newest bits. This operation scales as $\Omega(M^2)$, and can easily become the most time consuming operation of the algorithm (sec. 8.6). Note that if M is relatively small, the Efficient Trellis sigma-delta modulation algorithm is still more expensive than the other algorithms, but the difference is not very large. In this case the computational efficiency of the Pruned Tree sigma-delta modulation algorithm also approaches that of the Trellis sigma-delta modulation algorithm, since very limited sorting is required.

To simplify the comparison of the different look-ahead approaches their properties are summarized in table 11.1 in a qualitative matter. In this table the load per path reflects the number of operations required per path. Less operations per path is better. The average look-ahead depth per path is an indication of how many samples look-ahead is obtained, on average, per path. The more look-ahead there is realized, the better the performance of the algorithm. Finally, the scalability

of the computational load is an indication of the computational cost required to obtain more look-ahead. For example, in the case of a full look-ahead SDM the computational load doubles when the amount of look-ahead is increased by one, but in the case of a Pruned Tree SDM the computational load can be increased in a linear fashion.

	load per path	avg. LA depth per path	scalability of load
full look-ahead SDM	++	-	- -
Trellis SDM	+	+/-	- -
Efficient Trellis SDM	- -	+	+
Pruned Tree SDM	+/-	++	++
Pruned Tree SDM SA-CD	-	++	++

Table 11.1: Qualitative comparison of the different look-ahead approaches, ranging from very good (++) to very bad (- -).

11.3 Functional performance comparison

The functional performance of the full look-ahead sigma-delta modulation algorithm, the Trellis sigma-delta modulation algorithm, the Efficient Trellis sigma-delta modulation algorithm, the Pruned Tree sigma-delta modulation algorithm, and the Pruned Tree sigma-delta modulation for SA-CD look-ahead algorithm will be compared as a function of the number of parallel paths. Since the computational load per path varies between the different approaches, especially when a large number of paths is used, it is not possible to directly derive the performance per CPU cycle from the results, as explained in the previous section. However, if a small number of paths is used the computational load per path is almost equal for all the approaches, and as a result less paths means a smaller computational load, independent of the type of look-ahead technique.

11.3.1 SNR, SINAD, THD and SFDR

From the previous chapters it is known that the application of look-ahead techniques has a different impact on modulators that have a loop filter with resonator sections and modulators that have a loop filter without resonator sections. As a result, in order to compare the signal conversion performance of the different look-ahead approaches, it is necessary to

compare the performance for both type of filters separately. The same representative loop-filter configurations as in the previous chapters are used, i.e. SDM configuration SDM1, a fifth order loop filter without resonator sections, and configuration SDM2, a fifth order loop filter with resonators (see app. B).

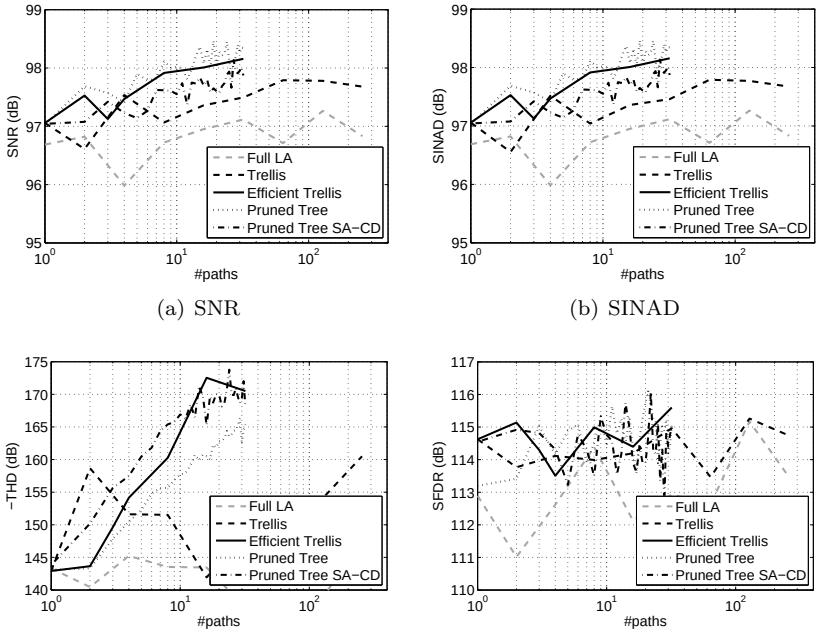
As demonstrated in ch. 7, it is sufficient to characterize the signal conversion performance for a single input frequency point, since the conversion performance is basically insensitive to the frequency of the input signal. However, since a typical SDM will generate harmonic distortion components it is necessary to use a low input frequency, such that all distortion components fall in the signal band and are considered in the THD and SFDR measures. By selecting a relatively large input amplitude for the characterization procedure, i.e. a difficult to convert signal, the difference between the conversion quality of the different algorithms will be most clearly visible.

Similar to the approach followed in the previous chapters, the SNR, the SINAD, the THD, and the SFDR performance that is obtained for a 1 kHz sine wave with an amplitude of 0.5 will be studied and compared, as a function of the number of parallel paths. In the case of the Trellis SDM the number of paths is dictated by the Trellis order N and is equal to 2^N. The amount of look-ahead realized by the full look-ahead sigma-delta modulation algorithm is also specified with N, with the number of paths present to realize this look-ahead equal to 2^{N-1}. In the case of the two Pruned Tree SDM realizations the number of paths can be freely selected. However, in the case of the Efficient Trellis SDM there is besides the parameter M that selects the number of parallel paths also the parameter N that has an impact on the realized performance. In the plot the best performance is depicted, which is obtained by selecting the optimal value of N for the selected number of paths.

SDM1

For loop-filter configuration SDM1 in fig. 11.1 the SNR (a), the SINAD (b), the THD (c), and the SFDR (d) are plotted as a function of the number of parallel paths for the different look-ahead techniques. In the Efficient Trellis SDM case the optimal value of N as a function of the selected number of paths is equal to $N = 8$ for $M \leq 4$ and $N = 16$ for more than four parallel paths.

With all the look-ahead techniques, except for the full look-ahead SDM, the SNR and the SINAD improve slightly when the number of paths is increased. The Efficient Trellis sigma-delta modulation algorithm and



(a) SNR (b) SINAD

(c) THD (d) SFDR

Figure 11.1: The SNR (a), the SINAD (b), the THD (c), and the SFDR (d) performance as a function of the number of parallel paths for SDM loop-filter configuration SDM1.

the normal Pruned Tree sigma-delta modulation algorithm realize the highest SNR and SINAD. The prediction filter to improve the lossless data compression gain has a negative impact on the SNR, and results in SNR and SINAD values that are approximately 0.4 dB lower. The Trellis sigma-delta modulation algorithm always results in SNR and SINAD values that are lower than what is achieved by the other pruned look-ahead algorithms, but better than what is achieved by the full look-ahead SDM, independent of the number of paths used. However, it has to be realized that the differences in the SNR values are not of a big relevance, since they all within 1.5 dB over the complete range of parallel paths.

The THD curves do show a big improvement as a function of the number of paths for all the look-ahead techniques, except for the full look-ahead SDM and the Trellis SDM. The Efficient Trellis SDM and the Pruned Tree SDM for SA-CD are the most effective in improving the THD, while the normal Pruned Tree SDM realizes an improvement of approximately

5 dB less if the same number of paths is used.

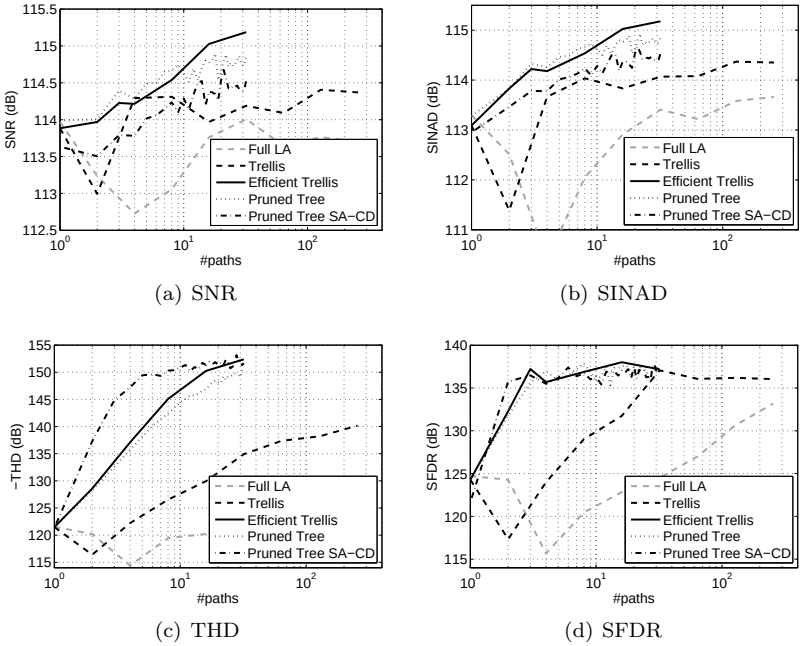
None of the look-ahead techniques is able to improve the SFDR for loop-filter configuration SDM1. This is not surprising since the SFDR is limited by the quantization noise at the end of the pass-band, as shown before in fig. 8.6, and since the SNR is virtually unchanged when more paths are used the noise level does not decrease.

Overall, with loop-filter configuration SDM1 the Efficient Trellis sigma-delta modulation algorithm is able to deliver the highest conversion quality since it realizes the maximum SNR and is able to significantly improve the THD with relatively few paths. The Pruned Tree sigma-delta modulation algorithm achieves the same SNR for the same number of parallel paths, but requires approximately three times more paths to achieve the same THD performance. However, the cost per path is much less so the difference in computational load will be very small. At the cost of a very minor penalty in the SNR, the Pruned Tree sigma-delta modulation algorithm for SA-CD is an interesting alternative since it realizes the biggest improvement in the THD with the least amount of paths, and requires far less computations per path than the Efficient Trellis sigma-delta modulation algorithm. The full look-ahead sigma-delta modulation and the Trellis sigma-delta modulation algorithm are a bad choice since they realize a lower SNR value and are hardly able to improve the THD.

SDM2

For loop-filter configuration SDM2 the value of N is selected as $N = 8$ for $M \leq 4$, $N = 16$ for $M = 8$ and $M = 16$, and $N = 32$ for $M = 32$. The shape of the SNR and SINAD curves (fig. 11.2(a) and (b)) is virtually identical to those of loop-filter configuration SDM1. The Efficient Trellis SDM and the Pruned Tree SDM realize the same high values, followed by the Pruned Tree SDM for SA-CD, while the full look-ahead SDM and the Trellis SDM realize the lowest values. As before, the improvement in the SNR and SINAD values is very minimal, i.e. less than 2 dB, and it is not considered to be relevant since much bigger improvements can be realized by changing the loop-filter corner frequency.

The THD performance, shown in fig. 11.2(c), improves the fastest for the Pruned Tree SDM for SA-CD. Next is the Efficient Trellis SDM that is, when the best value of N is used, slightly more effective than the normal Pruned Tree SDM. When 32 paths are used all three approaches result in approximately the same THD, that is more than 10 dB better than what is achieved by the Trellis SDM for 256 paths. Different from the



(a) SNR (b) SINAD

(c) THD (d) SFDR

Figure 11.2: The SNR (a), the SINAD (b), the THD (c), and the SFDR (d) performance as a function of the number of parallel paths for SDM loop-filter configuration SDM2.

situation with SDM configuration SDM1, application of the Trellis SDM now results in a steady improvement of the THD, but far less than what is realized by the other pruned look-ahead approaches. The full look-ahead sigma-delta modulation algorithm is not able to bring a significant reduction of the THD.

If no look-ahead is used, the SFDR for loop-filter configuration SDM2 is limited by harmonic distortion components (see fig. 8.9). However, since the look-ahead techniques, except for the full look-ahead sigma-delta modulation and Trellis sigma-delta modulation algorithm, are very effective in suppressing the distortion, already for a few paths the SFDR is not limited anymore by the harmonic distortion components but by the quantization noise. As a result, all the look-ahead techniques realize the same improvement in the SFDR, as shown in fig. 11.2(d).

The best overall performance for loop-filter configuration SDM2 is realized by the Pruned Tree sigma-delta modulation algorithm, i.e. it realizes

the same maximum SNR and the same improvement in the THD as the Efficient Trellis sigma-delta modulation algorithm for the same number of parallel paths, but it requires less computations than the Efficient Trellis sigma-delta modulation algorithm. At the cost of a very minor penalty in the SNR, the Pruned Tree sigma-delta modulation algorithm for SA-CD is an interesting alternative since it realizes the biggest improvement in the THD with the least amount of paths. Application of the full look-ahead sigma-delta modulation and Trellis sigma-delta modulation algorithms should not be preferred since the improvement of the THD and SNR is very minor compared to what the other look-ahead algorithms achieve while the computational load is much higher.

11.3.2 Converter stability

The stability of the different look-ahead modulators, as a function of the number of parallel paths, is investigated and evaluated using two different indicators. The first stability measure is the maximum amplitude of a 1 kHz sine wave that can be applied to the modulator without causing instability to the modulator. For the second stability test a 1 kHz sine wave with an amplitude of -6 dB is applied to the modulator and the maximum corner frequency of the loop filter that does not cause instability is determined. In both the experiments the loop filter has two resonator sections, equal to those of loop-filter configuration SDM2 (see app. B).

Maximum stable input amplitude

The maximum amplitude of a 1 kHz sine wave that can be applied to each type of look-ahead modulator is determined as a function of the number of parallel paths. The Efficient Trellis SDM is configured with $N = 16$ since this provides the maximum stability. The results of this experiment are depicted in fig. 11.3.

For a small number of paths the same stability is achieved with all the look-ahead modulators, except the full look-ahead SDM and the Trellis SDM that are significantly less stable. However, from 16 paths onwards the stability of the Efficient Trellis SDM does not improve anymore, while the two Pruned Tree SDM modulators do still become more stable. No difference can be detected between the normal Pruned Tree SDM and the Pruned Tree SDM for SA-CD. Thus, in order to support large signal amplitudes with minimal computational load the Pruned Tree sigma-delta modulation algorithm is the best choice.

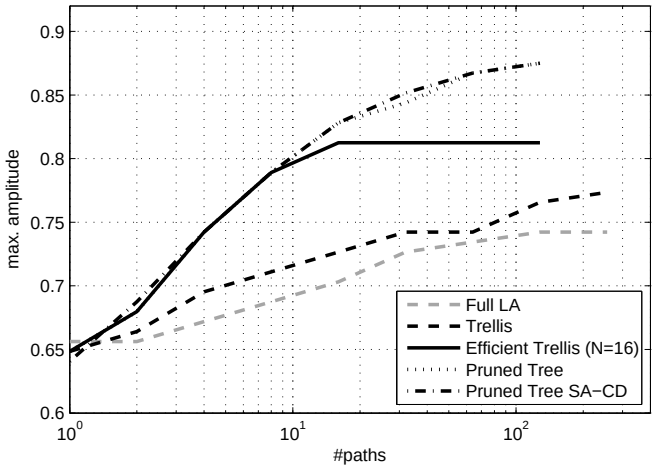


Figure 11.3: Maximum input amplitude that can be handled as a function of the number of parallel paths. All modulators are setup with loop-filter configuration SDM2.

Note that although the maximum amplitude that can be handled increases with the number of parallel paths, the maximum SNR that can be realized by the modulators does not increase proportionally. More specifically, for loop-filter configuration SDM2 the maximum SNR is realized for an amplitude of approximately 0.7, independent of the look-ahead technique and the number of paths used, as demonstrated in the previous chapters and illustrated in fig. 11.4 (Pruned Tree SDM with 128 paths).

Thus, although the maximum signal that can be handled without instability can be larger than 0.7, the maximum SNR is always realized for an amplitude of 0.7. The further the signal amplitude is above 0.7, the lower the SNR value will become. The reason for this phenomenon is the following. The number of bit sequences that can describe a signal reduces with the amplitude of the signal. If many possibilities exist it is possible to generate a bit sequence that describes the signal accurately, i.e. the signal is encoded with minimal quantization noise. If the number of possibilities reduces the quantization noise will increase. As long as the signal power increases more than the power of the quantization noise when the signal level is increased the SNR improves. This is the situation for signals up to a level of 0.7. However, from an amplitude

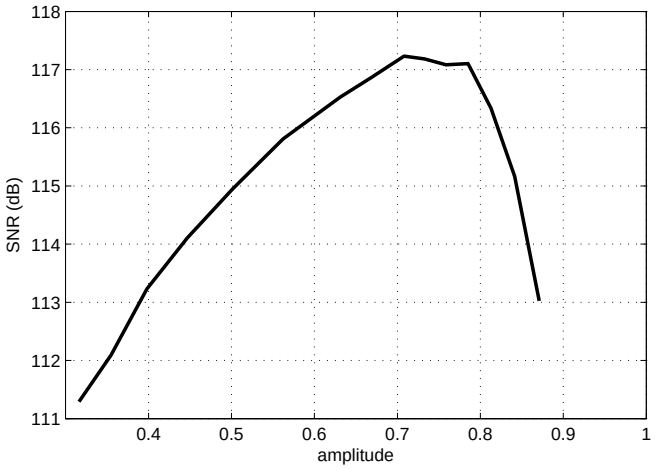


Figure 11.4: The SNR with loop-filter configuration SDM2 for large input amplitudes. The curve was generated with a Pruned Tree SDM with M=128.

of 0.8 onwards the number of possibilities to encode the signal reduces so fast that the quantization noise increases more than the signal power and a reduction of the SNR is resulting.

Maximum loop-filter corner frequency

In fig. 11.5 the maximum loop-filter corner frequency that can be used with a -6 dB 1 kHz input signal is depicted for the various look-ahead modulator types as a function of the number of parallel paths. Again, the Efficient Trellis SDM is configured with $N = 16$ to provide maximum stability.

The results are very similar to those found for the maximum signal amplitude test. More specifically, for a small number of parallel paths the Efficient Trellis SDM and the two Pruned Tree Sigma-Delta Modulators perform identical. The full look-ahead SDM as well as the Trellis SDM realize far less stability than the other modulators for the same number of parallel paths. The Efficient Trellis SDM does not realize any significant improvement in stability for more than 64 parallel paths. However, for both of the two Pruned Tree Sigma-Delta Modulators the stability keeps increasing when the number of paths is increased, making them a

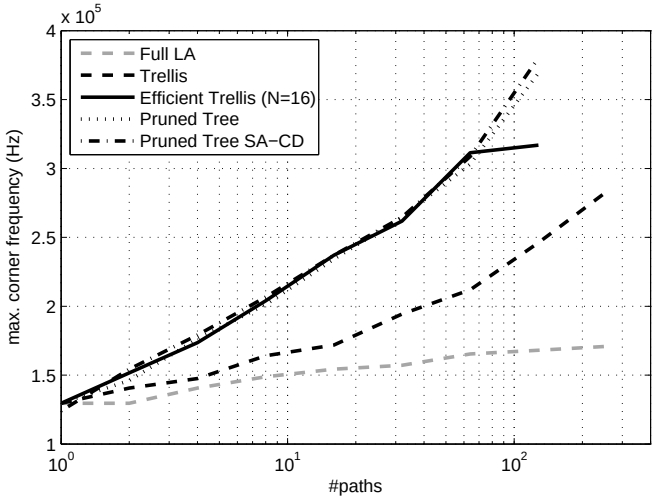


Figure 11.5: Maximum loop-filter corner frequency that can be used with a 1 kHz-6 dB input signal as a function of the number of parallel paths. The loop filter incorporates the resonator sections from loop-filter configuration SDM2.

much better choice.

In the previous chapters it has been shown that an increase of the corner frequency of the loop filter, initially, results in an increase of the SNR. However, once the corner-frequency reaches approximately 250 to 300 kHz the SNR stabilizes and a further increase of the corner frequency does not result in a higher SNR. This effect is demonstrated in fig. 11.6 for a Pruned Tree SDM with 128 parallel paths and a -6 dB input signal. This phenomenon occurs with all the look-ahead modulators and is investigated in detail in chapter 12.

11.3.3 Noise modulation

In the previous chapters the amount of noise modulation was, typically, investigated by measuring the in-band noise power as a function of the DC input level. However, a comparison of the quality of the different look-ahead techniques using this method is difficult. Furthermore, as shown in ch. 10, a characterization on the basis of DC input signals is not always the most appropriate one. The alternative procedure, i.e.

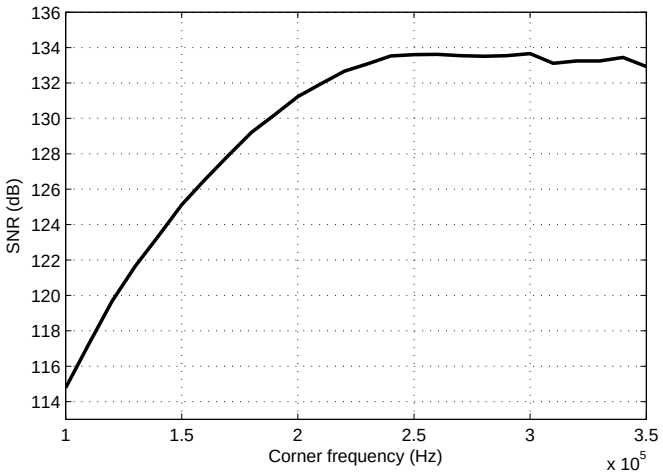


Figure 11.6: The obtained SNR as a function of the loop-filter corner frequency for a 1 kHz −6 dB input signal. The curve was generated with a Pruned Tree SDM with M=128.

a SINAD measurement as a function of the input level, also gives an indication of the amount of in-band noise, and can be conveniently used to compare the quality of the different look-ahead techniques. If the amount of in-band noise is constant for each amplitude level, i.e. there is no noise modulation, the SINAD value will increase linearly with the input amplitude. Thus, in this ideal case the SINAD versus input level plot will be a straight line. The more the SINAD versus input level curve deviates from this straight line, the more in-band noise modulation there is.

To compare the quality of the different look-ahead approaches they are all configured for maximum quality. More specifically, the full look-ahead SDM is set up to look-ahead 9 samples (256 parallel paths), the Trellis SDM is set up with $N = 8$ (256 parallel paths), the Efficient Trellis SDM with $N = 16$ and $M = 16$, the Pruned Tree SDM has $M = 16$ parallel paths, and the Pruned Tree SDM for SA-CD has only $M = 4$ parallel paths. In the case of the Pruned Tree SDM for SA-CD no significant change in performance is detected when more paths are used. The SINAD of a 1 kHz input signal is measured for loop-filter configuration SDM2 (app. B) for levels ranging between -120 dB and -3 dB. The results of the experiment are depicted in fig. 11.7.

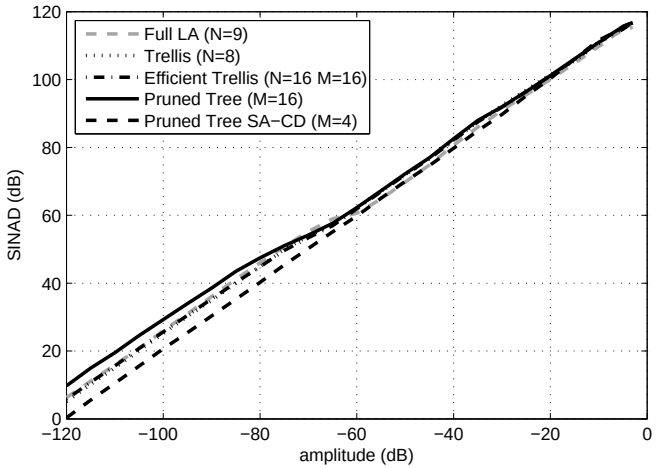


Figure 11.7: SINAD as a function of the input amplitude for the various look-ahead techniques.

From the five different look-ahead approaches the Pruned Tree SDM realizes the highest SINAD for low input amplitudes and the Pruned Tree SDM for SA-CD the lowest. The SINAD of the Trellis SDM and the Efficient Trellis SDM is equal and is in between that of the two Pruned Tree SDM look-ahead techniques. The curve of the full look-ahead SDM approximately follows that of the Trellis SDM for low level inputs and then shifts to the curve of the Pruned Tree SDM for SA-CD. At an input level of approximately -70 dB the SINAD curves of the Trellis SDM, the Efficient Trellis SDM, and the Pruned Tree SDM have a downward bend and come together. At an input level of approximately -30 dB the three curves have another bend and shift to the same level as the curve of the Pruned Tree SDM for SA-CD. In the case of the Pruned Tree SDM for SA-CD the SINAD increases virtually linearly with the input level and no significant noise modulation is present. The normal Pruned Tree SDM has the largest amount of noise modulation, while the full look-ahead SDM, the Trellis SDM, and the Efficient Trellis SDM are slightly better.

In the case of an audio focussed application noise modulation is considered problematic. However, in most other applications noise modulation is not considered as a problem, and the focus is on realizing an SNR as high as possible under all conditions. In this situation the Pruned

Tree sigma-delta modulation algorithm is the preferred algorithm since it realizes, especially for low amplitude signals, the highest SNR.

The reason that the Pruned Tree sigma-delta modulation algorithm realizes a higher SNR for the low amplitude signals than the other algorithms is the following. All look-ahead algorithms try to maximize the SNR of the encoded signal. For large amplitude signals they all realize the same SNR since there are few possibilities to encode the signal. However, for low amplitude signals there are many possibilities to encode the signal, and depending on the selected solution a higher SNR can be resulting. Thus, if an algorithm searches a larger relevant portion of the solution space the probability for realizing a higher SNR will increase. From the stability experiments it is known that the two Pruned Tree sigma-delta modulation algorithms realize a larger stability than the other algorithms, which indicates that a larger relevant portion of the solution space is searched by the algorithms. The normal Pruned Tree sigma-delta modulation algorithm has as only optimization criterion the realization of a maximum SNR, and this is resulting in the high SNR for low amplitude signals. The Pruned Tree sigma-delta modulation algorithm for SA-CD takes also the predictability of the output bitstream into account. The effect of this second optimization criterion is that a minimal, nearly constant, amount of additional noise is present in the output signal which results in the lower SNR for low amplitude signals. When the signal amplitude increases, the impact of this additional noise on the SNR is negligible and the same SNR is obtained as with the other look-ahead techniques.

11.3.4 Lossless data compression

The average lossless data compression gain that can be obtained on a bitstream depends on the signal characteristics of the SDM encoded signal, but also on the loop-filter design of the modulator that was used to create the bitstream, and on the look-ahead technique used. For the various look-ahead techniques, as a function of the number of parallel paths, the compression gain that is obtained for an encoded 1 kHz-60 dB sine wave is measured. In the experiment, instead of applying the actual very computational intensive SA-CD DST compression algorithm, the compression gain is calculated on the basis of the data size reduction realized by the traditional (fast) ZIP algorithm, known from the popular ".ZIP" file format [48]. These results are therefore a first order indication of the actual compression gain that will be achieved by the DST compression algorithm, although they are slightly optimistic.

More specifically, the DST algorithm is not only designed to give good compression gains for 1-bit encoded data, but it is also designed such that the decoding process can be implemented efficiently in hardware. As a result, the DST algorithm will, in general, not be able to match the compression performance of the ZIP algorithm that can make use of extensive hardware resources. Furthermore, the quantization noise spectrum that is resulting from actual music signals is slightly more difficult to predict than the spectrum resulting from sinusoids, and will consequently result in a lower compression gain. For the generation of the input bitstreams loop-filter configuration SDM2 is used (app. B). The Efficient Trellis SDM is configured with $N = 32$, since this results in a slightly higher compression gain than $N = 16$.

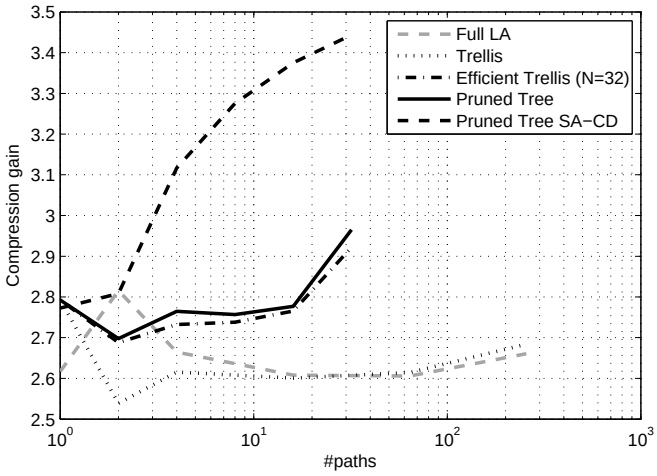


Figure 11.8: Lossless data compression gain for a -60 dB 1 kHz sine wave as a function of the number of parallel paths for the various look-ahead techniques. The compression gain has been measured with the ZIP algorithm.

The results of the experiment are depicted in fig. 11.8. The compression gain obtained by the Trellis sigma-delta modulation and the full look-ahead sigma-delta modulation algorithm are significantly lower than that of the other look-ahead techniques, and it increases only slowly with the number of parallel paths. The performance of the Efficient Trellis SDM and the Pruned Tree SDM is nearly identical, and virtually constant for 16 or less paths. When the number of paths is increased from 16 to 32 the compression gain increases significantly, resulting in

a total increase of 6.2% compared to the situation with a single path. Compared to the other look-ahead techniques the Pruned Tree SDM for SA-CD is much more effective in realizing a high compression gain. When four paths are used an increase of 12.4% is realized, and with 32 paths a total increase of 24% in compression gain is realized.

As mentioned before, real life audio signals are, typically, more difficult to compress than single sine waves. Thus, from the results in fig. 11.8 it is not possible to make accurate estimates for the compression gain that will be achieved for real audio data when compressed with the DST algorithm. In order to investigate this further, for three different audio recordings, i.e. a classical piece, a jazz recording, and a pop track, the average compression gain that is achieved by the DST compression algorithm has been measured for various look-ahead configurations. More specifically, for a realization of the Pruned Tree sigma-delta modulation algorithm and for the Pruned Tree sigma-delta modulation algorithm for SA-CD, both with the same loop filter, the average compression gain as a function of the number of parallel paths has been derived. As a reference point, the compression gain for a normal SDM with the same loop filter has also been measured. The results of this experiment are depicted in fig. 11.9.

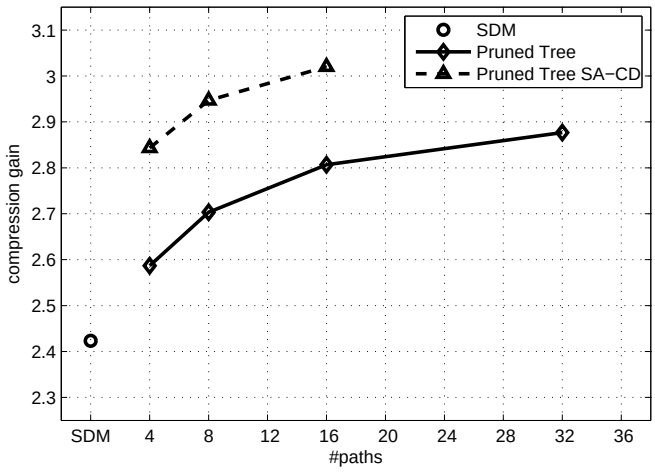


Figure 11.9: Comparison of the average lossless data compression gain for actual musical content. The compression gain has been measured with the DST algorithm.

In the first experiment where the compression gain was measured of sine waves, the compression gain of the Pruned Tree SDM generated bitstream was hardly influenced by the number of parallel paths, and only in the case of 32 parallel paths an improvement in the compression gain was realized. In the case of the measurements on actual audio data, in general a lower compression gain is realized, but now an increase of the compression gain is realized when more paths are used. As before, the compression gain of the Pruned Tree SDM for SA-CD is significantly higher than that of the normal Pruned Tree SDM, and also here a strong increase in the compression gain is resulting when the number of parallel paths is increased. Compared to the normal SDM, a compression gain of more than 20% higher can be realized by the Pruned Tree SDM for SA-CD with only eight parallel paths. In the case of the normal Pruned Tree SDM more than 32 parallel paths are required to reach this performance.

11.3.5 Summary

In general, the signal conversion performance of the full look-ahead sigma-delta modulation algorithm is of a lower quality than that of the other look-ahead techniques, even if many more paths are used. The Trellis sigma-delta modulation algorithm performs a bit better, but still much worse than the other pruned look-ahead approaches. The difference in performance between the Efficient Trellis sigma-delta modulation algorithm, the Pruned Tree sigma-delta modulation algorithm, and the Pruned Tree sigma-delta modulation algorithm for SA-CD is relatively small. The three algorithms realize approximately the same SNR and SINAD when the same number of paths are used. If minimization of the THD at a minimum computational cost is desired, the Pruned Tree sigma-delta modulation algorithm for SA-CD is clearly the best choice. However, if more parallel paths are used, for example in order to improve the stability of the converter, the THD improvement realized by the different algorithms becomes the same. In this case the Efficient Trellis sigma-delta modulation algorithm should not be preferred since it results in a larger computational load than the two Pruned Tree sigma-delta modulation algorithms.

If it is desired to realize a converter that is maximally stable, e.g. to support a large input range or use aggressive noise shaping, the Pruned Tree sigma-delta modulation based algorithms are the best choice, since they always perform equal or better than the Efficient Trellis sigma-delta modulation algorithm and require less computations per output sample. The stability of the full look-ahead sigma-delta modulation and

Trellis sigma-delta modulation algorithm is far less than that of the other look-ahead techniques.

In terms of noise modulation the Pruned Tree sigma-delta modulation algorithm for SA-CD is delivering the best performance, and realizes a virtually perfect SINAD versus amplitude curve with as little as four parallel paths. The normal Pruned Tree sigma-delta modulation algorithm delivers the worst noise modulation performance. However, if maximization of the SNR for small input signals is important, the Pruned Tree sigma-delta modulation algorithm is the best choice.

The Pruned Tree sigma-delta modulation algorithm for SA-CD was specifically designed to generate bitstreams that can be compressed well. With respect to the other look-ahead techniques the algorithm indeed outperforms them by a very large amount, and a good scalability as a function of the number of parallel paths is achieved. The bitstreams generated by the normal Pruned Tree sigma-delta modulation and the Efficient Trellis sigma-delta modulation algorithm have a compression gain that is virtually not affected by the number of parallel paths, and only in the case a large amount of paths is used the compression gain increases slightly. The bitstreams generated by the Trellis sigma-delta modulation and the full look-ahead sigma-delta modulation algorithm are the most difficult to compress, and even with 128 parallel paths the compression gain is lower than what is achieved by the other algorithms with a single path.

Overall, the two Pruned Tree sigma-delta modulation algorithms deliver the highest signal conversion quality, are the most stable, and require the least amount of computational resources. If it is not important to realize bitstreams with a good audio quality, i.e. noise modulation is acceptable and lossless data compression is not relevant, the normal Pruned Tree sigma-delta modulation algorithm is the best choice since it delivers the highest SNR for low amplitude signals. However, if the bitstreams are going to be used for SA-CD mastering purposes, the Pruned Tree sigma-delta modulation algorithm for SA-CD is the best choice.

11.4 Conclusions

An analysis has been made of the alternative look-ahead techniques that are described in literature. The result of this investigation is that most of the techniques are already covered by the work described in the previous chapters. For the other described techniques it was found that they will always deliver less performance than the in this work studied algorithms,

or that it is not clear from the information in the publications how the approach should be extended to high order loop filters on which the focus of this work is. As a result, in this chapter only a comparison was made between the in this work described look-ahead techniques.

From an algorithmic point of view the Trellis sigma-delta modulation algorithm, the Efficient Trellis sigma-delta modulation algorithm, and the Pruned Tree sigma-delta modulation algorithm (for SA-CD) are very similar. The main difference between the algorithms is the selection cost function. All the other steps of the algorithms are nearly identical, with small differences that are resulting from the way how the paths that continue are selected. However, there is a large impact of the selection cost function on the computational load and the signal conversion performance of the algorithms. The Pruned Tree sigma-delta modulation algorithm for SA-CD is equal to the normal Pruned Tree sigma-delta modulation algorithm from an algorithmic point of view, but is based on a different path cost function that takes into account the predictability of the resulting bitstream.

In general, the performance of the full look-ahead sigma-delta modulation algorithm is the worst from all the five look-ahead algorithms, i.e. it requires the most computations and delivers the smallest improvement in signal conversion performance. The pruned look-ahead approaches all realize more performance, although the Trellis sigma-delta modulation algorithm is only minimally more effective than the full look-ahead sigma-delta modulation algorithm. The Pruned Tree sigma-delta modulation algorithm for SA-CD delivers on most aspects the best performance with the lowest number of paths.

Independent of the look-ahead algorithm, the SNR performance is almost insensitive to the number of parallel paths and is approximately the same for all the algorithms. The THD performance, typically, improves with the number of parallel paths, most efficiently with the Pruned Tree sigma-delta modulation algorithm for SA-CD. For example, with only four parallel paths an improvement of the THD of 25 dB is realized. In the case of a loop filter with resonator sections also the SFDR will improve significantly because of the reduced level of the signal harmonics. If there are no resonator sections the SFDR will be limited by the noise at the end of the signal band, and the reduction of the THD will only result in an increase of the SINAD.

From all the studied look-ahead algorithms the normal Pruned Tree sigma-delta modulation algorithm is able to deliver the largest stability at the minimal computational load. The stability of the Pruned Tree sigma-delta modulation algorithm for SA-CD increases exactly the same

with the number of paths, but the algorithm requires slightly more computations per path because of the more complex cost function that takes the predictability of the bitstream into account. The increase in stability can be used to convert larger signals or to enable more aggressive noise shaping. For example, by increasing the number of paths to 128 the input range for SDM configuration SDM2 can be enlarged by 35% compared to the situation with one path. Although in some specific cases, e.g. for SA-CD mastering purposes, such an increase of the input range can be desirable, it does not result in a significant increase of the SNR. In fact, independent of the look-ahead technique, with SDM configuration SDM2 the maximum SNR is realized for a signal level of 0.7. Therefore, it is more interesting to use the increased stability to apply more aggressive noise shaping that can, if used properly, result in significantly improved SNR values.

Independent of the noise-shaping characteristic, all the studied look-ahead algorithms realize approximately the same SNR for large input signals as long as the system is stable. However, for low signal levels there is a large deviation in the SNR. As a result, the noise modulation performance of the algorithms differs significantly. The Pruned Tree SDM algorithm realizes for low amplitudes an SNR that is 10 dB higher than expected on the basis of the SNR that is obtained for high amplitude signals, and suffers the most from noise modulation. The Pruned Tree sigma-delta modulation algorithm for SA-CD delivers the best noise modulation performance and realizes a perfect linear SNR vs. input amplitude transfer. The performance of the full look-ahead sigma-delta modulation, the Trellis sigma-delta modulation and the Efficient Trellis sigma-delta modulation algorithms is approximately the same and is in between that of the other two algorithms.

In terms of loss-less data compression compatibility the Pruned Tree sigma-delta modulation algorithm for SA-CD outperforms the other algorithms by far. Results on actual music recordings show that the Pruned Tree sigma-delta modulation algorithm for SA-CD with four parallel paths delivers almost the same performance as the normal Pruned Tree sigma-delta modulation algorithm with 32 paths, and compared to a normal SDM an improvement of more than 20% is realized. By increasing the number of paths to 16 the playback time can be enlarged by almost 25% compared to a normal SDM.

All in all, for Super Audio CD mastering applications the Pruned Tree sigma-delta modulation algorithm for SA-CD is clearly preferred over the other look-ahead techniques. It provides the best audio encoding properties, i.e. low distortion and no noise modulation, and it is able

to generate bitstreams that can be compressed well, resulting in a large maximum playback duration. If the objective is to have a modulator that realizes the maximum SNR for every input level at a minimal computational load the normal Pruned Tree sigma-delta modulation algorithm is the best choice.

Chapter 12

Maximum SNR analysis

In the previous chapters it was observed that when the corner frequency of the loop filter is increased the SNR initially increases but that from some corner frequency onwards the SNR does not improve anymore. This result is against expectations and is investigated in more detail in this chapter. In sec. 12.1 a first experiment is described that explores the SNR limits as a function of the loop-filter corner frequency for different signal levels. In a second experiment, presented in sec. 12.2, the SNR development as a function of the loop-filter corner frequency for different filter orders is investigated. An analysis of the experimental results is made in sec. 12.3. The outcome, a theory that predicts the loop-filter corner frequency that results in the optimal noise shaping, is used to maximize the SNR of a look-ahead converter in sec. 12.4. In sec. 12.5 the obtained SNR values are compared to the theoretical maximum obtainable SNR and compared to the practically required maximum SNR. Finally, conclusions are drawn in sec. 12.6

12.1 Experiment 1

An experiment is performed in which for different signal levels, as a function of the corner frequency of the loop filter, the SNR of a 1 kHz sinusoid is measured. The loop filter is a fifth order filter with two resonator sections, similar to loop-filter configuration SDM2 (app. B). In the experiment a Pruned Tree SDM is used with the number of parallel paths constant at 128. The loop-filter corner frequency is increased in steps of 10 kHz from 100 kHz to 500 kHz. The results of this experiment are depicted in fig. 12.1.

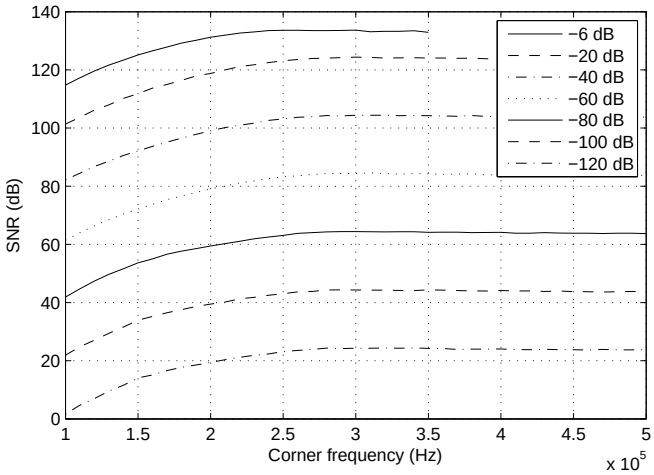
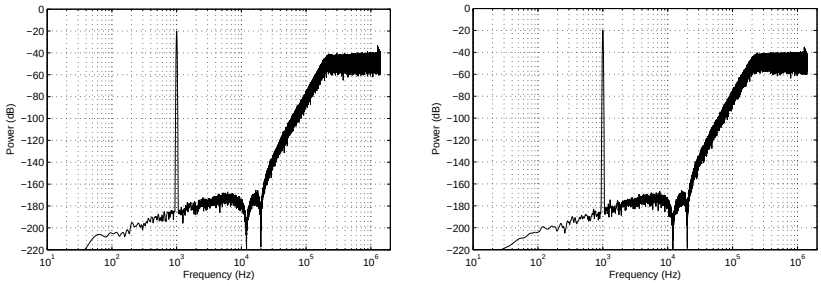


Figure 12.1: SNR as a function of the loop-filter corner frequency and the signal level for 5th order loop filter with resonator sections. The Pruned Tree SDM is configured with 128 parallel paths.

Independent of the signal level, from a loop-filter corner frequency of approximately 300 kHz onwards the SNR does not increase anymore. A close inspection reveals that there is a small dependency of the signal level on the corner frequency for which the maximum SNR is realized, i.e. for all signals with a level of -20 dB or less the maximum SNR is realized for 300 kHz while for the -6 dB input signal the maximum SNR is realized already for approximately 250 kHz. If the corner frequency is increased above the point where the maximum SNR is realized a minimal reduction in the SNR is resulting. This reduction is less than 0.5 dB for the low input signals and about 1 dB for the -20 dB input signal, and is therefore hardly visible in the figure. Note that for the -6 dB signal the converter is only stable for corner frequencies up to 350 kHz, while for the lower level input signals the converter is stable to at least 500 kHz.

In order to understand why the SNR does not increase anymore when the corner frequency is increased above 300 kHz, the output spectrum for the -20 dB input signal is compared between a loop filter with 300 kHz corner frequency and one with 400 kHz corner frequency. The two spectra are shown in fig. 12.2. Although the two loop filters are very different, clearly the spectra are virtually equal. Thus, independent of the noise-shaping filter the same noise shaping is realized. Investigations show

that the output spectrum is virtually equal for every loop filter with a corner frequency above 300 kHz. However, although the application of a higher corner frequency does not result in a different noise-shaping characteristic or higher SNR, it does result in a system that is less stable. This reduction in stability is the reason for the minimal reduction in SNR for the higher corner frequencies.



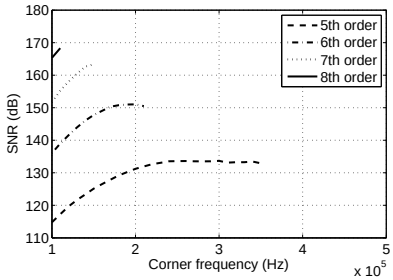
(a) 300 kHz (b) 400 kHz

Figure 12.2: Output spectrum for a 1 kHz -20 dB input for a fifth order loop filter with a corner frequency of 300 kHz (a) and for a fifth order loop filter with a corner frequency of 400 kHz (b). The Pruned Tree SDM is configured with 128 parallel paths.

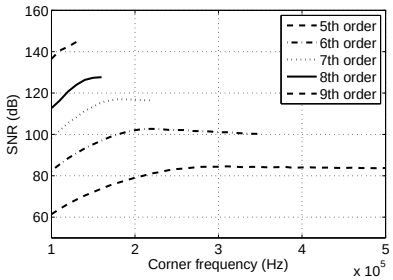
The spectra in fig. 12.2 show another interesting phenomenon. The typical output spectrum of a 1-bit SDM shows strong tonal behavior in the high frequency region. These tones are correlated with the input signal [50]. However, in the case of a loop filter with a corner frequency of 300 kHz or more there are hardly any high frequency tones, i.e. the high frequency part of the spectrum is nearly flat except for some minimal tones near $fs/2$.

12.2 Experiment 2

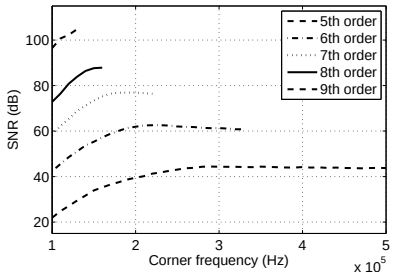
In order to get more insight in the reason why the noise-shaping characteristics become constant above a certain loop-filter corner frequency, the first experiment is repeated, but now for different filter orders. In fig. 12.3 the result is shown for three different input levels, i.e.-6 dB, -60 dB, and -100 dB, for filter orders of five till nine. For each filter order the number of resonators and the location of the notches have been optimized to give the highest SNR possible.



(a) -6 dB



(b) -60 dB



(c) -100 dB

Figure 12.3: SNR as a function of the loop-filter corner frequency for fifth till ninth order loop filters, for an amplitude of -6 dB (a), -60 dB (b), and -100 dB (c). In all cases the Pruned Tree SDM is configured with 128 parallel paths. The ninth order filter can not be stabilized for the -6 dB input level.

From the figure it is clear that, as expected, an increase of the filter order results in a higher SNR for the same corner frequency. It is also clear that a higher filter order results in less stability, visible in the form of a reduced maximum corner frequency that can be used without causing instability to the system. For all filter orders there is a different specific frequency for which the maximum SNR is obtained. Higher corner frequencies again result in a reduction of the SNR, which is clearly visible for the higher filter orders. The corner frequency that results in the maximum SNR reduces with the filter order, e.g. from 300 kHz for the fifth order filter to 200 kHz for the seventh order filter. The maximum corner frequency that can be used for the -100 dB signal is only minimally higher than what can be used for the -60 dB signal. In the case of the -6 dB a strong reduction in the maximum corner frequency compared to the low amplitude signals is resulting, and the

system is not stable for the ninth order filter with a corner frequency of 100 kHz.

12.3 Analysis

Analysis of the results shown above reveals that for the corner frequency that results in the maximum SNR the loop-filter feed-forward coefficients have special values. More specifically, for low amplitude signals the maximum SNR is realized at approximately the corner frequency for which the first two feed-forward coefficients of the loop filter, i.e. $b(1)$ and $b(2)$ in fig. 12.4, are equal.

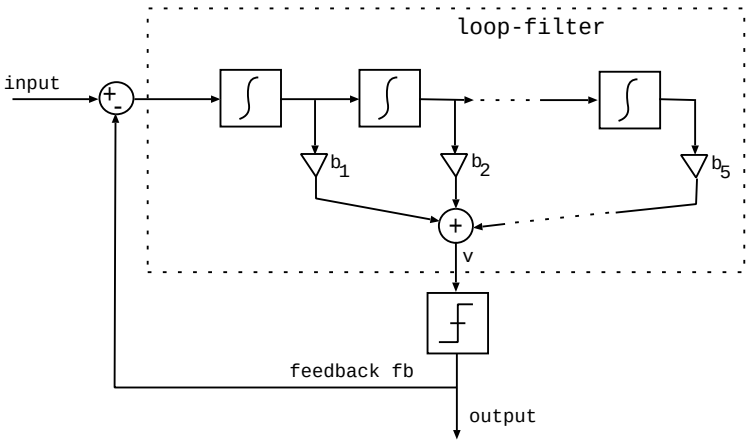


Figure 12.4: Fifth order feed-forward SDM.

Table 12.1 lists the corner frequencies at which the maximum SNR is observed and compares this with the corner frequencies for which it holds that $b(1) = b(2)$. Corner frequencies below the point of the maximum SNR have $b(1) > b(2)$, and filters with a higher corner frequency have $b(1) < b(2)$. In the case of the -6 dB signal the maximum SNR is, typically, realized for a slightly lower corner frequency. This can be attributed to a reduced stability of the system due to the large signal. Note that for all amplitude levels the SNR increases only minimally between a corner frequency of some tens of kHz below the optimal point and the actual optimal point while the stability is affected significantly in this range.

filter order	observed frequency (kHz)	calculated frequency (kHz)
5	290	302
6	230	246
7	200	208
8	160[1]	181
9	130[1]	160

Table 12.1: The observed corner frequency that results in the maximum SNR and the corner frequency that results in loop-filter coefficients $b(1) = b(2)$. [1]This is the maximum corner frequency that can be used with 128 parallel paths without causing instability.

The explanation for the saturation of the noise-shaping system towards the point where $b(1) = b(2)$ is the following. At the point where $b(1) = b(2)$ the noise-shaping system is marginally stable, and the maximum possible noise shaping is realized. Higher corner frequencies are stabilized by the look-ahead algorithm and will result in the same feedback signal that would be generated for the marginally stable situation. As a result, in these cases the same noise shaping will be realized as for the marginally stable filter.

Because the noise shaping look-ahead system is highly non-linear it is very difficult to proof that a loop filter with $b(1) = b(2)$ will result in a marginally stable system that results in the maximum possible noise shaping. Therefore, this will not be attempted, but instead it will be shown that this theory is plausible.

12.3.1 Second order filter stability

To get more insight in the crossover point from marginally stable to unstable, we will first study the simple linear second order system of fig. 12.5.

Consider the situation that $b(1) = b(2) = 1$. In this case the loop-filter transfer (without feed-back) is $H = \frac{z^{-1}}{1-2z^{-1}+z^{-2}}$. Solving for the pole locations of the closed-loop system results in $|z| = 1$, independent of the feedback gain. Thus, this system is always marginally stable.

Now consider the same second order linear feedback system with $b(1) < b(2)$. For example, consider a setup with $b(1) = 1, b(2) = 2$. The transfer function of the loop filter is $H = \frac{z^{-1}+z^{-2}}{1-2z^{-1}+z^{-2}}$. In the case of

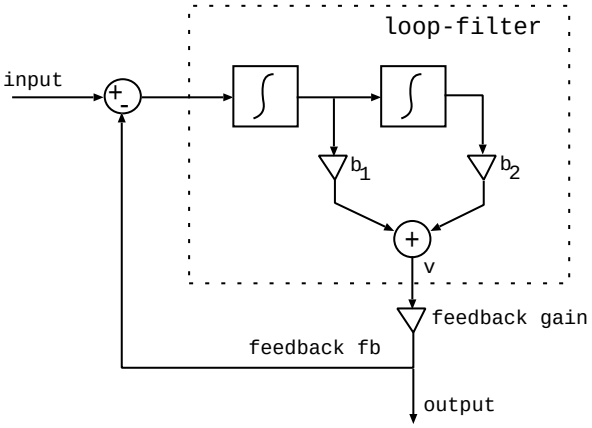


Figure 12.5: Second order linear system with feedback.

a unity feed-back gain the poles of the system have $|z| = \sqrt{2}$, and the system is unstable. In fact, the system is always unstable, independent of the feed-back gain.

A 1-bit SDM is not a simple linear feed-back system and can, typically, not be compared one-to-one to the linear feed-back system. However, if we replace the quantizer with a linear (effective) gain it is possible to study the stability of the linearized feed-back system. The problem with this approach is that the conclusion about the stability of the system is only valid as long as the assumed effective quantizer gain is correct. Since the effective quantizer gain is a function of the input signal and the loop filter this method is not very reliable, and incorrect conclusions can be drawn about the stability of the SDM. However, in some specific cases the assessment of the stability of the feed-back system can be made without approximation. In the case of the second order system discussed above it is clear that with $b(1) = b(2) = 1$ the feedback system is always stable, independent of the feedback gain. Thus, if this filter is used in a normal 1-bit SDM the system is also stable. In the situation where $b(1) < b(2)$ the linear feedback system is always unstable. As a result, a normal 1-bit SDM will also be unstable with this filter. However, if look-ahead is added to this second order SDM with the unstable filter, the system will become stable and the same noise shaping will be realized as for the marginally stable situation.

First consider the normal 1-bit SDM of fig. 12.6 with the filter $b(1) = b(2) = 1$, and a constant zero input signal. In table 12.2 the content of

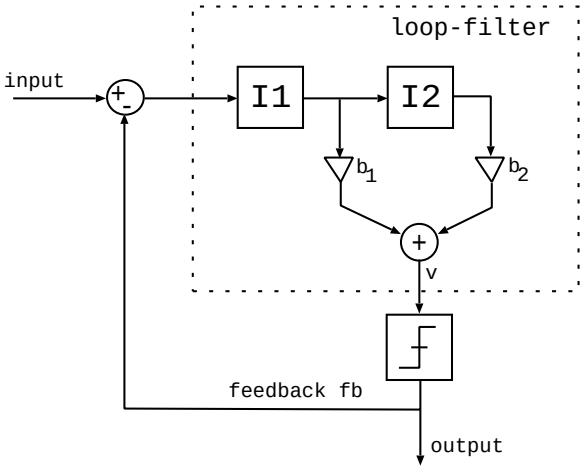


Figure 12.6: Second order 1-bit SDM.

the loop-filter integrators I1 and I2, the input to the quantizer Qin, and the output of the quantizer $Qout$ is listed for ten clock cycles. Clearly the integrator values stay small, and the output is toggling at a high rate, which indicates that, as predicted, the system is stable.

cycle	I1	I2	Qin	$Qout$
1	0	0	0	-1
2	1	0	1	1
3	0	1	1	1
4	-1	1	0	-1
5	0	0	0	-1
6	1	0	1	1
7	0	1	1	1
8	-1	1	0	-1
9	0	0	0	-1
10	1	0	1	1

Table 12.2: Internal states I1 and I2, quantizer input Qin and quantizer output $Qout$ for a normal 1-bit SDM with (stable) loop-filter coefficients $b(1) = b(2) = 1$. The system is marginally stable.

Now consider a normal 1-bit SDM with the filter $b(1) = 1$, $b(2) = 2$. The time domain results for this filter are tabulated in table 12.3. As

expected, the feed-back strategy of the SDM results in a system that is unstable, recognizable by the constantly growing values of the internal integrators and the feed-back signal with a decreasing frequency.

cycle	I1	I2	Qin	$Qout$
1	0	0	0	-1
2	1	0	1	1
3	0	1	2	1
4	-1	1	1	1
5	-2	0	-2	-1
6	-1	-2	-5	-1
7	0	-3	-6	-1
8	1	-3	-5	-1
9	2	-2	-2	-1
10	3	0	3	1

Table 12.3: Internal states I1 and I2, quantizer input Qin and quantizer output $Qout$ for a normal 1-bit SDM with (unstable) loop-filter coefficients $b(1) = 1$, $b(2) = 2$. The system is unstable.

It will now be demonstrated that application of look-ahead can stabilize the system with $b(1) = 1$, $b(2) = 2$. Similar to the look-ahead algorithms described earlier, the look-ahead algorithm will try to minimize the sum of the filter output v squared, i.e. $\sum v^2$, by selecting the proper feedback symbol fb. The result of this experiment is listed in table 12.4. The internal states of the system stay small and the feedback signal does not show a low frequency oscillation. Thus, a feedback sequence that results in stable operation of the system has been found.

Note that the filter output v is comparable to the quantizer input Qin, and that the feedback value fb is comparable to the quantizer output $Qout$. Comparison of the signal fb of table 12.4 to the signal $Qout$ of table 12.2 reveals that the two signals are identical. Thus, the look-ahead algorithm has stabilized the second order unstable filter by generating the same feedback signal as what is generated with the normal feed-back strategy for the marginally stable filter.

12.3.2 High order filter stability

In the case of a 1-bit SDM with a high order filter it is not possible to calculate exactly when the system becomes unstable. Only by assuming an effective quantizer gain it can be calculated if the system is stable or

cycle	I1	I2	v	fb
1	0	0	0	-1
2	1	0	1	1
3	0	1	2	1
4	-1	1	1	-1
5	0	0	0	-1
6	1	0	1	1
7	0	1	2	1
8	-1	1	1	-1
9	0	0	0	-1
10	1	0	1	1

Table 12.4: Internal states I1 and I2, filter output v and feedback value fb for a look-ahead 1-bit SDM with (unstable) loop-filter coefficients $b(1) = 1, b(2) = 2$. The signal fb results in a stable system.

not. However, the effective quantizer gain depends on the input signal and the loop filter, and can only be approximated. If the assumed effective quantizer gain is incorrect the calculated point of instability will also be incorrect. Thus, it is not possible to calculate with good accuracy the corner frequency for which the system will become unstable on the basis of the linear model.

As an alternative to assuming an effective quantizer gain in order to calculate the corner frequency for which the SDM will become unstable, it is possible to derive the required effective quantizer gain that will result in a critically stable system for the corner frequency that results in the maximum SNR under the assumption that the required effective quantizer gain is a constant. The outcome of these calculations is that an effective quantizer gain of 0.735 will result in corner frequencies that are very close to the observed frequencies that result in the maximum SNR. The corner frequencies that are obtained are listed in table 12.5 together with the observed corner frequency that results in the maximum SNR.

The estimate of the corner frequency that results in the maximum SNR that is based on an effective quantizer gain of 0.735 matches the observed frequency that results in the maximum SNR to a very large extent. Furthermore, the derived effective quantizer gain is a realistic value since it indicates that the quantizer input is, typically, larger than one, which is expected for a system that is close to instability.

A comparison of the corner frequencies calculated by solving the corner frequency that results in $b(1) = b(2)$ (table 12.1) and the frequencies

filter order	observed frequency (kHz)	calculated frequency (kHz)
5	290	300
6	230	237
7	200	202
8	160[1]	178
9	130[1]	159

Table 12.5: The observed corner frequency that results in the maximum SNR and the maximum corner frequency that results in marginally stable operation under the assumption of an effective quantizer gain of 0.735. [1]This is the maximum corner frequency that can be used with 128 parallel paths without causing instability.

calculated by assuming a critically stable system for an effective quantizer gain of 0.735 (table 12.5) shows that the two calculations are in good agreement and that both match well with the observed frequencies that result in the maximum SNR. From this it can be concluded that it is very probable that the corner frequency that results in the maximum SNR is indeed the point at which the feedback system is critically stable. Increasing the corner frequency above this point will result in an, in principle, non-stable feed-back system that will be stabilized by the look-ahead algorithm to the point of marginal stability. As a result, the SNR will decrease slightly compared to the optimal point because of the reduced stability of the system. If a lower corner frequency is used a less than optimal noise shaping will be realized. However, if a slightly lower corner frequency is selected, a minimal penalty on the obtainable SNR is resulting, but the stability of the converter is significantly improved, resulting in a lower number of required parallel paths.

12.4 Obtaining the maximum SNR

From the above presented results it can be seen that in order to obtain the maximum SNR for a given computational load it is, typically, more effective to increase the filter order than to increase the filter corner frequency. More specifically, increasing the corner frequency from 100 kHz to the point of the maximum SNR will result in an increase of approximately 20 dB, independent of the filter order. Increasing the filter order by one while keeping the corner frequency at 100 kHz will also result in an SNR increase of approximately 20 dB. However, if the number of

parallel paths is high enough to result in stable operation at the corner frequency that results in the maximum SNR, the system is, typically, also stable enough to increase the filter order by approximately two while keeping a corner frequency of 100 kHz. Thus, by increasing the filter order by two and keeping the corner frequency at 100 kHz an improvement of approximately 40 dB in the SNR can be obtained, while increasing the corner frequency from 100 kHz to the point of maximum SNR will only give an improvement of around 20 dB. Note that it is possible to use loop-filter corner frequencies slightly below 100 kHz with high order filters but that in this case, typically, an undesirable peaking occurs at the corner frequency. From 100 kHz onwards this phenomenon reduces substantially, resulting in a smooth spectrum.

As an example, consider the maximum SNR that can be obtained with 128 parallel paths. With this number of parallel paths it is possible to use an eighth order loop filter with a corner frequency of 110 kHz while supporting signals as large as -6 dB (fig. 12.3). The resulting SNR for this configuration is 168 dB. If a seventh order filter is used instead, the maximum corner frequency that can be used is 150 kHz which results in an SNR of 163 dB. If it is desired to improve significantly on this result the number of parallel paths has to be increased such that stable operation of a ninth order filter will become possible. By running the Pruned Tree SDM with 512 parallel paths this is possible and an SNR of 190 dB is obtained for a loop-filter corner frequency of 110 kHz. The output spectrum is shown in fig. 12.7. If the filter order is kept at eight and the filter order is maximally increased to the point just before instability, a maximum SNR of only 173 dB is achieved for the filter with a corner frequency of 130 kHz. Thus, for the same computational complexity, again the higher order filter results in a higher SNR.

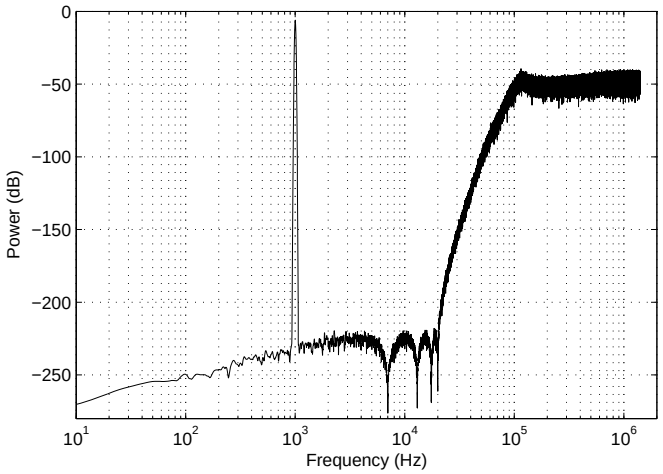


Figure 12.7: Output spectrum of a Pruned Tree SDM (M=512) with a ninth order loop filter with a corner frequency of 110 kHz. The input signal is a 1 kHz sine wave with an amplitude of −6 dB. The SNR equals 190 dB.

12.5 Theoretical maximum SNR

From the Shannon-Hartley theorem we know that the channel capacity is given by:

$$C = B \cdot log_2(1 + \frac{S}{N}) \quad (12.1)$$

$$= 2 \cdot B \cdot log_2(M) \quad (12.2)$$

where C is the channel capacity in bits per second, B is the bandwidth of the channel in hertz, S/N is the signal-to-noise power ratio, and M is the number of distinguishable levels.

In the case of a 64-times oversampled 1-bit SDM with a sampling rate of approximately 2.8 MHz, the channel capacity is:

$$C = 64 \cdot 44\,100 \cdot 1$$

$$= 2\,822\,400 \; bits \; per \; second \quad (12.3)$$

The theorem states that the maximum obtainable SNR, calculated over the complete frequency band, is limited to 4.77 dB. Only by considering

the signal and the noise power over a fraction of the frequency band a higher SNR can be obtained. The highest SNR over the baseband region will be obtained if all the quantization noise is shifted to high frequencies.

If the complete channel capacity could be allocated to describe a signal in a bandwidth of 22.05 kHz very accurately, and have all the noise outside this region, a maximum SNR of 385 dB could be obtained, which is equal to the expected equivalent of 64-bit PCM. In the experiments in this thesis the SNR is calculated over the slightly lower bandwidth of 20 kHz. Over this bandwidth, if the complete channel capacity could be allocated, the maximum possible SNR would be 425 dB, which is the equivalent of 70-bit PCM.

The example ninth-order modulator from sec. 12.4 that achieves an SNR of 190 dB, is only using $\frac{190}{425} = 44.7\%$ of the total channel capacity. In order to utilize more of the channel capacity, a higher baseband SNR is required, which can be realized by using a higher loop-filter order.

With the theory from sec. 12.3 it is possible to calculate the loop-filter corner frequency that results in a critically stable system, for any filter order. This corner frequency that results in the optimal (maximal) noise shaping reduces with the filter order. In order to avoid strong spectral peaking at the corner frequency, a phenomenon that is present especially for high order loop filters, and to have reasonable constraints for the reconstruction low-pass filter, a practical lower limit on the corner frequency of the loop filter is 100 kHz. Calculation of the corner frequency that results in critically stable operation for filter orders above nine, reveals that with a 14th order loop filter the optimal noise shaping is achieved for a corner frequency of 102 kHz. In the case of a 15th order loop filter the optimal corner frequency reduces to 95 kHz, and becomes impractically low. An extrapolation from the SNR results obtained with the 9th order filter, indicates that the 14th order loop filter would generate an SNR of approximately 290 dB. In this case 68% of the channel capacity would be used. By selecting a lower loop-filter corner frequency, in combination with a higher order filter, it should be possible to obtain an even higher SNR. However, the practical usefulness of such a noise-shaping characteristic is very limited, since a very steep low-pass reconstruction filter would be required.

Although the discussion above is interesting from a theoretical point of view, it is also insightful to compare the results with the practical demands on the SNR. If it is desired to realize a 1-bit modulator that is completely transparent, i.e. to not degrade the SINAD of the original signal significantly, the SNR of the modulator should be 20 dB higher

than that of the original input signal. If it is assumed that the input signal has an SNR equivalent[1] of 120 dB, and that only one quantization operation is performed, the SNR of the 1-bit modulator should be around 140 dB. If during editing or mastering a signal is re-quantized multiple times, a procedure that should ideally be avoided but that is practice in some recording studios, the SNR requirements on the modulator are slightly higher because the quantization noise is added multiple times. Depending on the number of re-quantizations that are performed, the target SNR can be calculated. If it is assumed that no more than 10 re-quantizations are performed, an SNR of 150 dB is sufficient to realize transparent encoding. In both cases a sixth order look-ahead modulator can provide this level of performance (sec. 12.2). However, in most situations the dynamic range of signals is limited on purpose to allow comfortable listening at reduced playback levels, and the equivalent SNR of the input signal is in the order of only 100 dB. In this case an SNR of 120 dB, or 130 dB if re-quantizations will be performed, is adequate, and a fifth order look-ahead modulator can be used.

By combining the above results and the results from sec. 12.4 it can be concluded that, in practice, the maximum SNR that can be obtained with a look-ahead modulator is not limited from an information theory point of view, but only by the amount of available computational resources that are required to stabilize the very aggressive high order filters. If the complete recording and processing chain is considered, there is little need for the extremely high SNR ratios that could be realized if enough computational resources would be available, and even the most demanding recording situation can be supported with a 150 dB SNR modulator.

12.6 Conclusions

The traditional approach to increase the SNR of a 1-bit SDM is to increase the loop-filter corner frequency. However, it has been demonstrated that there is a limit of what can be achieved with this approach. More specifically, in the case of a high order loop filter there is a point from which increasing the corner frequency further does not result in an increase of the SNR, i.e. there is a point of maximal noise shaping. Because of the non-linear behavior of a 1-bit SDM it is difficult to proof that this specific loop-filter corner frequency results in a critically stable closed-loop noise-shaping system, and therefore it has only been made

[1] the SNR of a microphone is typically significantly less than 100 dB, but the dynamic range can be more than 120 dB

plausible by analyzing the stability of a second order system. From the experimental results on high order loop filters it has been derived that the corner frequency that results in a critically stable system can be found by searching for the loop-filter corner frequency for which the first two coefficients of the loop filter become equal, or by solving for critical stability of the linearized noise-shaping loop with an effective quantizer gain of 0.735.

Operation of an SDM at the point of maximal noise shaping has a severe penalty on the stability of the SDM. By selecting a slightly lower loop-filter corner frequency than the frequency that results in the maximal noise shaping, a small penalty in the SNR is resulting, but a significant increase in the stability of the converter is obtained. Use of a higher corner frequency does, in principle, result in the same SNR since the look-ahead algorithm will stabilize the system to the point of critical stability. However, in practice, a lower than maximal SNR will be resulting because of the significant reduction of the stability of the noise shaper.

A loop filter design procedure to realize the maximum SNR for a given computational load has been derived. First, the maximum filter order where a corner frequency of at least 100 kHz can be used with the maximum desired signal level should be selected. Second, for the selected filter order the highest corner frequency that results in stable operation should be determined. If this procedure is followed the selected corner frequency will always be below the point where the system becomes critically stable, otherwise a higher filter order could be used. With this procedure a record SNR of 190 dB was realized by configuring a Pruned Tree SDM with 512 paths and a ninth order loop filter with a corner frequency of 110 kHz.

Finally, it has been shown that, in practice, the maximum obtainable SNR is not limited by the channel capacity, but only by the available amount of computational resources. From a practical point of view there is, even for the most demanding audio applications, no need to realize an SNR of more than 150 dB. Because of the limits on the achievable amount of noise shaping, at least a sixth order loop filter is required to realize such an SNR value. If a more reasonable maximum SNR of 130 dB is desired, i.e. an SNR that still enables virtually lossless 1-bit audio encoding in most practical situations, a fifth order loop filter can be used. In this case, if the converter is configured as an Efficient Trellis SDM or as a Pruned Tree SDM, only eight parallel paths are required to stabilize the converter, and a practically very feasible solution is resulting.

Chapter 13

General conclusions

The main operations of a look-ahead SDM can be summarized as the parallel investigation of a number of potential encoding solutions, the evaluation of their quality, and the selection of the output symbol. Since this requires many identical realizations of the same loop filter, look-ahead techniques can not be easily applied to analog-to-digital conversion, but they can bring large benefits to digital-to-digital 1-bit sigma-delta modulation.

It has been demonstrated that the traditional full look-ahead approach is not computationally efficient and that only minimal improvements in signal conversion quality, compared to a normal SDM, can be obtained. By pruning the solution space, i.e. removing solutions that will most likely not contribute to the final solution, it is possible to realize a look-ahead modulator with a high computational efficiency. Because of this a larger amount of look-ahead can be obtained and a higher signal conversion quality will be resulting.

The Trellis sigma-delta modulation algorithm, an improvement on the full look-ahead algorithm that performs a limited amount of pruning, has been analyzed. It realizes a higher signal conversion quality at a reduced computational load, but still the approach is too expensive to be practically usable. It was recognized that only a fraction of all the solutions under investigation are contributing to the final output, which resulted in the conception of the Efficient Trellis sigma-delta modulation algorithm. The aggressive pruning of the solution space reduces the computational load significantly and enables a larger pruned look-ahead depth. Simulations have shown that such an approach pays off and that a more linear SDM with better stability is resulting. In the Pruned

Tree sigma-delta modulation algorithm the constraints on the pruning have been relaxed, and a more efficient solution has been realized that delivers a comparable signal quality.

Typical improvements of a Pruned Tree SDM over a normal SDM are an increase of the linearity, an increase of the stability, and a reduction of the amount of noise modulation, all simultaneously realized and scalable with the number of parallel paths. The increase in stability can be used to support larger input signals or to allow for more aggressive noise shaping.

It has further been demonstrated that, with a large enough number of parallel paths, it is possible to stabilize converters with an unstable noise-shaping loop. The noise shaping that results in this case is equal to that obtained for the critically stable situation. This point of maximal noise shaping is a function of the loop-filter order and can be calculated on the basis of the linear model. A further outcome of the analysis is that the maximum SNR that can be obtained with 1-bit sigma-delta modulation is depending on the loop-filter order, and that the key to maximizing the SNR is to use a filter order as high as possible instead of maximizing the loop-filter corner frequency.

Finally, in order to come to a look-ahead modulator that is suitable for SA-CD applications, the Pruned Tree sigma-delta modulation algorithm has been extended with a prediction cost function. This modification results in a modulator that has all the benefits of the normal Pruned Tree SDM, but that is generating bitstreams with an increased level of predictability, which causes a higher lossless data compression gain. In addition to this, the prediction cost function, that acts as a signal dependent dither, results in a further improvement of the linearity of the modulator and eliminates all the noise modulation. Experiments on real music recordings show that with the Pruned Tree sigma-delta modulation algorithm for SA-CD an increase of the lossless compression gain of more than 20%, compared to a normal SDM, can be obtained without difficulties.

In summary, the pruning concept presented in this thesis is supported by several look-ahead modulator realizations and their performance evaluation. Both in the generic 1-bit look-ahead modulator case, and in the specific case of a 1-bit modulator for SA-CD, major improvements over state-of-the-art have been achieved.

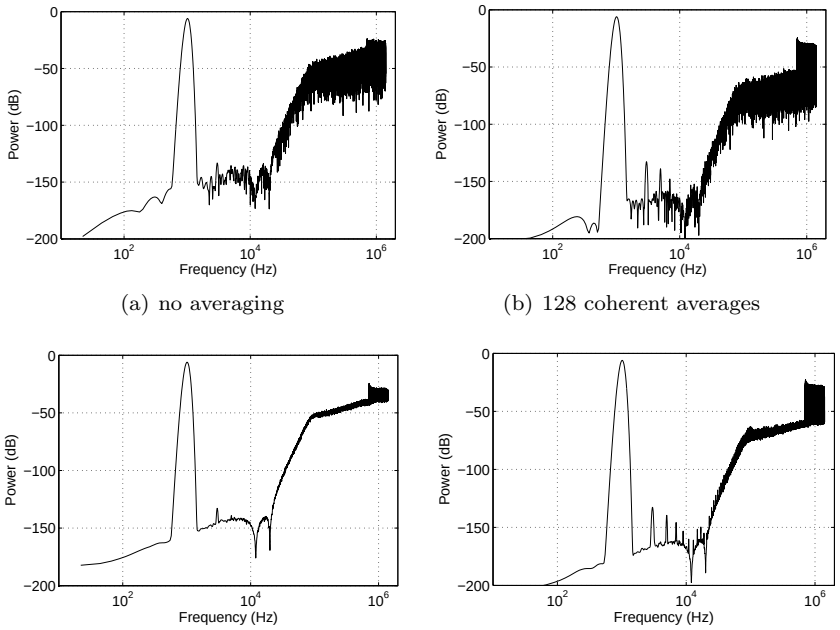
Appendix A

FFT calculations - coherent and power averaging

Because of the high linearity and the low noise levels of the look-ahead Sigma-Delta Modulators described in this thesis, a high accuracy spectral performance evaluation is required. This is achieved by performing a windowed FFT, in combination with power averaging and coherent averaging. The FFT window used is a Gaussian window, such that it is possible to obtain the exact power and frequency of each spectral tone, also if the frequency of the tone is not equal to the FFT bin frequency. The effect of power and coherent averaging, techniques not generally known, is explained below.

In the case of coherent averaging, the averaging is performed in the time domain. The SDM under investigation performs a conversion of the same (identical) signal N times, with the only difference between the N conversions the initial conditions of the converter. The time domain output of the modulator will describe the same signal N times, with the only difference between the N signals resulting from the different quantization noise. The result of averaging the time domain signal is that the power of signal components that are identical for each of the conversion runs will not be affected, but the average power of signals components that are varying from run to run will be reduced. If signals are uncorrelated, e.g. ideal quantization noise, the average power will reduce with 3 dB for every doubling of the number of averages. Thus, for analysis purposes the noise floor of the SDM output can be lowered by an arbitrary amount by performing enough coherent averages, such that distortion tones which are below the noise floor become visible. Note that the output after coherent averaging can not be used to calculate

the SNR, the SINAD, or the SFDR. Compare fig. A.1(a) and (b) for the effect of 128 coherent averages.



(a) no averaging

(b) 128 coherent averages

(c) 32 power averages

(d) 128 coherent and 32 power averages

Figure A.1: The effect of coherent averaging and power averaging. The FFT length is $128 \cdot 10^3$ samples.

In the process of power averaging, the averaging is performed in the spectral domain. More specifically, the average is taken over M evaluations of the power spectrum, obtained from M output sequences of the SDM. Since the average is performed in the power domain, the phase of the signal is not relevant, and the only requirement on the SDM conversion is that not both the input signal and the initial state of the converter are identical from run to run. If either the phase of the signal or the initial conditions of the converter are varying, the power of the signal components will be identical but the noise power will vary. By averaging the power spectra the average power spectral density is obtained, which results in a smooth spectral plot. Power averaging has no effect on the total amount of noise power. The difference between 32 power averages and no power averaging can be clearly seen by comparing fig. A.1(a) and (c).

If coherent and power averaging are combined, the power averaging is performed over the coherently averaged signals. Thus, for each of the M power averages, N coherent averages need to be performed. The total number of output sequences that need to be generated by the SDM is equal to $N \cdot M$. The number of FFT evaluations is equal to M. The result of a combination of coherent and power averaging is a smooth spectrum with a reduced quantization noise floor level, such that the low level distortion tones can be clearly identified. Fig. A.1(d) shows the result of 32 power averages in combination with 128 coherent averages.

Appendix B

Description of the used Sigma-Delta Modulators

Throughout this thesis a number of SDM loop-filter configurations are regularly used. The configurations are either named "SDMx" or "SDMxFB", where the latter is the feed-back version of the former feed-forward configuration, and the "x" denotes the configuration number. In all the cases a sampling rate of $64 \cdot 44\,100$ Hz (approximately 2.8 MHz) is assumed. Table B.1 lists the specifications of the configurations.

configuration	loop-filter order	corner frequency (kHz)	resonator frequency (kHz)
SDM1	5	100	-
SDM2	5	100	12, 20
SDM3	5	140	12, 20
SDM4	3	200	20

Table B.1: Specifications of the different loop-filter configurations, all for an assumed sampling rate of 2.8 MHz.

All loop filters are designed from a Butterworth prototype filter, according to the procedure described in [52]. The resonator sections are added afterwards and consist of a simple feed-back structure. In fig. B.1 the implementation structure of a fifth order feed-forward loop filter with resonator sections is depicted. The loop-filter coefficients of all the loop-filter configurations are listed in table B.2.

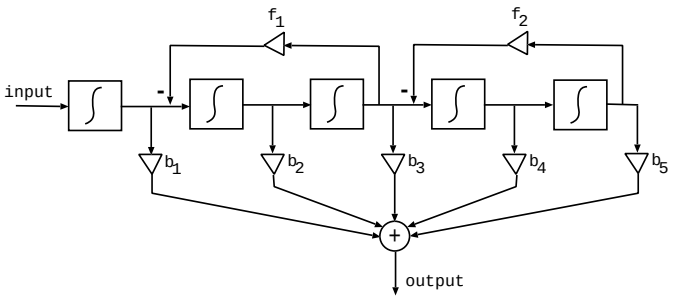


Figure B.1: Fifth order feed-forward loop filter with resonators.

configuration	b1	b2	b3	b4	b5	f1	f2
SDM1	0.7200	0.2524	0.0530	0.0066	0.0004	-	-
SDM2	0.7200	0.2524	0.0530	0.0066	0.0004	0.0007	0.0020
SDM3	1.0074	0.4890	0.1405	0.0236	0.0018	0.0007	0.0020
SDM4	0.8840	0.3505	0.0592	-	-	0.0020	-

Table B.2: Loop-filter coefficients of the different loop-filter configurations.

References

[1] R. M. Aarts, D. Reefman, and E. Janssen. Super audio CD - An overview. In *Proceedings of the 18th International Congress on Acoustics (ICA2004)*, volume 3, pages 2179–2182, April 2004. Paper No.We4.E.3, Kyoto, Japan. Invited paper.

[2] S. S. Abeysekera and C. Charoensak. Performance evaluation of 3rd order sigma-delta ($\Sigma - \Delta$) modulators via fpga implementation. In *ASIC/SOC Conference, 2001. Proceedings. 14th Annual IEEE International*, pages 13–17, 2001.

[3] J. A. S. Angus. Efficient algorithms for look-ahead sigma delta modulators. In *Proceedings of the AES 115'th convention*, October 2003. preprint 5950, October 10-13 New York.

[4] J. A. S. Angus. Tree based lookahead sigma delta modulators. In *Proceedings of the AES 114'th convention*, March 2003. preprint 5825, March 22-25 Amsterdam.

[5] J. A. S. Angus. Implementation of "tree" and "stack" algorithms for look-ahead sigma delta modulators. In *Proceedings of the AES 117'th convention*, October 2004. preprint 6281, October 28-31 San Francisco.

[6] J. A. S. Angus. A comparison of the "pruned-tree" versus "stack" algorithms for look-ahead sigma-delta modulators. *Journal of the Audio Engineering Society*, 54(6), June 2006.

[7] J. Candy. A use of limit cycle oscillations to obtain robust analog-to-digital converters. *IEEE Transactions on Communications*, 22:298–305, 1974.

[8] K. Chao, S. Nadeem, W. Lee, and C. Sodini. A higher order topology for interpolative modulators foroversampling A/D converters. *IEEE Transactions on Circuits and Systems*, 37(3):309–318, 1990.

[9] C. C. Cutler. Differential quantization of communication signals. U.S. Patent 2,605,361. filed June 29, 1950, issued July 29, 1952.

[10] C. C. Cutler. Transmission systems employing quantization. U.S. Patent 2,927,962. filed April 26, 1954, issued March 8, 1960.

[11] F. de Jager. Delta modulation - a method of PCM transmission using the one unit code. Rep. Vol. 7 pp. 442–466, Philips Research, 1952.

[12] E. M. Deloraine, S. V. Mierlo, and B. Derjavitch. Communication system utilizing constant amplitude pulses of opposite polarities. U.S. Patent 2,629,857. filed October 8, 1947, issued February 24, 1953.

[13] E. M. Deloraine, S. V. Mierlo, and B. Derjavitch. Methode et systéme de transmission par impulsions. French Patent 932,140. issued August, 1946.

[14] C. Dunn and M. Sandler. Psychoacoustically Optimal Sigma-Delta Modulation. *Journal of the Audio Engineering Society*, 45(4):212–223, 1997.

[15] R. Fano. A heuristic discussion of probabilistic decoding. *Information Theory, IEEE Transactions on*, 9:64–74, 1963.

[16] G. Forney. The viterbi algorithm. *Proceedings of the IEEE*, 61(3):268–278, 1973.

[17] D. J. Goodman. The application of delta modulation of analog-to-PCM encoding. *Bell System Technical Journal*, 48:321–343, February 1969. Reprinted in N. S. Jayant, Waveform Quantization and Coding, IEEE Press and John Wiley, 1976, ISBN 0-471-01970-4.

[18] G. C. Goodwin, D. E. Quevedo, and D. McGrath. Moving-horizon optimal quantizer for audio signals. *Journal of the Audio Engineering Society*, 51(3), March 2003.

[19] R. Gray and T. Stockham Jr. Dithered quantizers. *IEEE Transactions on Information Theory*, 39(3):805–812, 1993.

[20] P. Harpe, D. Reefman, and E. Janssen. Efficient Trellis-type Sigma-Delta Modulator. In *Proceedings of the AES 114'th convention*, 2003. Preprint 5845, March 22-25 Amsterdam.

[21] M. O. J. Hawksford. Parametrically controlled noise shaping in variable state-step-back pseudo-Trellis SDM. In *Proceedings of the AES 115th convention*, October 2003. preprint 5877 October 10-13 New York.

[22] M. O. J. Hawksford. Parametrically controlled noise shaping in variable state-step-back pseudo-Trellis SDM. *Vision, Image and Signal Processing, IEE Proceedings* -, 152(1):87–96, 2005.

[23] M. O. J. Hawksford. Energy balance decision threshold in SDM systems. In *31st AES Int. Conf. on High-Resolution Audio*, June 2007. London, June 25–27.

[24] M. O. J. Hawksford. Parallel Look-Ahead digital SDM with Energy-Balance binary comparator. *Journal of the Audio Engineering Society*, 56(12):1069, 2008.

[25] S. Hoare and J. A. S. Angus. The performance of look-ahead sigma-delta modulators with unstable noise shaping filters. In *Proceedings of the AES 121st convention*, October 2006. preprint 6865 October 5-8 San Francisco.

[26] H. Inose and Y. Yasuda. A unity bit coding method by negative feedback. *Proceedings of the IEEE*, 51:1524–1535, 1963.

[27] H. Inose, Y. Yasuda, and J. Murakami. A telemetering system by code modulation: $\Delta - \Sigma$ modulation. *IRE Transactions on Space Electronics Telemetry*, 8:204–209, September 1962. Reprinted in N. S. Jayant, Waveform Quantization and Coding, IEEE Press and John Wiley, 1976, ISBN 0-471-01970-4.

[28] E. Janssen, E. Knapen, D. Reefman, and F. Bruekers. Lossless compression of one-bit audio. In *Acoustics, Speech, and Signal Processing (ICASSP2004)*, 2004. May 17-41 Montreal. Invited paper.

[29] E. Janssen and D. Reefman. Adaptive filtering. U.S. Patent Application 2007/0052556 A1. filed Apr 15, 2004.

[30] E. Janssen and D. Reefman. Generating bit-streams with higher compression gains. U.S. Patent 7,218,263 B2. filed October 8, 2004, issued May 15, 2007.

[31] E. Janssen and D. Reefman. Sigma-delta modulator with a quantizer/gain element. U.S. Patent 7,330,142. filed Apr 13, 2004, issued Feb 12, 2008.

[32] E. Janssen and D. Reefman. Advances in Trellis based SDM structures. In *Proceedings of the AES 115'th convention*, 2003. Preprint 5993, October 10-13 New York.

[33] E. Janssen and D. Reefman. Super-audio CD: an introduction. In *IEEE Signal Processing Magazine*, volume 20, July 2003. Invited paper.

[34] E. Janssen and D. Reefman. DSD compression for recent ultra high quality 1-bit coders. In *Proceedings of the AES 118th convention*, 2005. preprint 6470 May 28-31 Barcelona.

[35] E. Janssen and A. H. M. v. Roermund. A performance comparison of look-ahead sigma-delta modulation algorithms. *IEEE Transactions on Circuits and Systems I*, 2010. Submitted.

[36] F. Jelinek. Fast sequential decoding algorithm using a stack. *IBM J. Res. Dev.*, 13(6):675–685, 1969.

[37] H. Kato. Trellis noise shaping converters architecture and evaluation. Technical Report 381, Technical report of IEICE. EA, 2001.

[38] H. Kato. Trellis noise-shaping convertors and 1-bit digital audio. In *Proceedings of the AES 112th convention*, May 2002. Preprint 5615, May 10-13 Munich.

[39] E. Knapen, D. Reefman, E. Janssen, and F. Bruekers. Lossless compression of one-bit audio. *Journal of the Audio Engineering Society*, 52(3):190–199, March 2004. Invited paper.

[40] D. Knuth. *The Art of Computer Programming*, volume 3: Sorting and Searching, Second Edition. Addison-Wesley, 1998. ISBN 0-201-89685-0. Section 5.2.1: Sorting by Insertion.

[41] S. Lipshitz, J. Vanderkooy, and R. Wannamaker. Minimally audible noise shaping. *Journal of the Audio Engineering Society*, 39(11):836–852, 1991.

[42] S. Lipshitz, R. Wannamaker, and J. Vanderkooy. Dithered noise shapers and recursive digital filters. *Journal of the Audio Engineering Society*, 52(11):1124–1142, November 2004.

[43] A. Magrath. *Algorithms and architectures for high resolution Sigma-Delta converters*. PhD thesis, University of London, 1996.

[44] A. Magrath and M. Sandler. Efficient dithering of sigma-delta modulators with adaptive bitflipping. *Electronics Letters*, 31(11):846–847, 1995.

[45] A. Magrath and M. Sandler. Non-linear deterministic dithering of sigma-delta modulators. In *IEE Colloquium on Oversampling and Sigma-Delta Strategies for DSP*, page 2, 1995.

[46] S. Norsworthy, R. Schreier, and G. Temes. *Delta-Sigma Converters, Theory, Design and Simulation.* IEEE Press, New York, 1997.

[47] Philips and Sony. *Super Audio CD System Description.* Philips licensing, Eindhoven, The Netherlands, 2002.

[48] PKWARE. The inventor of pkzip. `http://www.pkware.com/`.

[49] D. Reefman, P. Harpe, and E. Janssen. Noise-shaping device and method with improved lossless compression and good audio quality for high fidelity audio. U.S. Patent Application 2007/0290906 Al. filed July 15, 2004.

[50] D. Reefman and E. Janssen. DC analysis of high order sigma delta modulators. In *Proceedings of the AES 113th convention*, 2002. preprint 5693 October 5-8 Los Angeles.

[51] D. Reefman and E. Janssen. Enhanced Sigma Delta structures for Super Audio CD application. In *Proceedings of the AES 112'th convention*, 2002. preprint 5616, May 10-13 Munich.

[52] D. Reefman and E. Janssen. *Signal processing for Direct Stream Digital.* Philips IP&S, Available upon request, December 2002.

[53] D. Reefman and E. Janssen. One-bit Audio: An Overview. *Journal of the Audio Engineering Society*, 52(3):166–189, March 2004. Invited paper.

[54] D. Reefman, E. Janssen, J. D. Reiss, and M. B. Sandler. Improved compression of DSD for Super Audio CD. In *Proceedings of the AES 112th convention*, 2002. preprint 5617 May 10-13 Munich.

[55] D. Reefman, J. D. Reiss, E. Janssen, and M. B. Sandler. Stability analysis of limit cycles in high order sigma delta modulators. In *Proceedings of the AES 115th convention*, 2003. preprint 5936 October 10-13 New York.

[56] D. Reefman, J. D. Reiss, E. Janssen, and M. B. Sandler. Description of limit cycles in feedback Sigma Delta Modulators. In *Proceedings of the AES 117th convention*, 2004. preprint 6280 October 28-31 San Francisco.

[57] D. Reefman, J. D. Reiss, E. Janssen, and M. B. Sandler. Description of limit cycles in Sigma Delta Modulators. *IEEE Transactions on Circuits and Systems I*, 52:1211–1223, June 2005.

[58] L. Risbo. ΣΔ *Modulators Stability Analysis and Optimization.* PhD thesis, Technical University of Denmark, June 1994.

[59] R. Schreier. An empirical study of high-order single-bit delta-sigma modulators. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 40(8):461–466, 1993.

[60] J. T. Stonick, J. L. Rulla, and S. H. Ardalan. Look-ahead decision-feedback ΣΔ modulation. In *Acoustics, Speech, and Signal Processing (ICASSP), 1994*, volume 3, pages 541–544, 1994.

[61] J. Vanderkooy and S. Lipshitz. Dither in digital audio. *Journal of the Audio Engineering Society*, 35(12):966–975, 1987.

[62] A. Viterbi. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *Information Theory, IEEE Transactions on*, 13(2):260–269, 1967.

[63] R. Wannamaker. *The theory of dithered quantization.* PhD thesis, Department of Applied Mathematics, University of Waterloo, 1997.

[64] R. Wannamaker, S. Lipshitz, J. Vanderkooy, and J. Wright. A theory of nonsubtractive dither. *IEEE Transactions on Signal Processing*, 48(2):499–516, 2000.

[65] P. Websdell and J. A. S. Angus. The effect of different metrics on the performance of "stack" algorithms for look-ahead sigma delta modulators. In *Proceedings of the AES 123rd convention*, October 2007. preprint 7202 October 5-8 New York.

[66] Wikipedia. Sorting algorithm — wikipedia, the free encyclopedia, 2009. `http://en.wikipedia.org/w/index.php?title=Sorting_algorithm&oldid=322728960`.

Original contributions

- Development of a time-domain SINAD measurement method, enabling the evaluation of the SINAD performance of an SDM for non-steady-state signals.
- Derivation of a linear model that predicts the STF and NTF of a look-ahead SDM.
- Analysis of the possibilities to reduce the computational complexity of a look-ahead modulator, resulting in the pruned look-ahead approach.
- Analysis of the Trellis sigma-delta modulation algorithm and the impact of its design parameters on performance.
- Derivation, implementation, and performance evaluation of two computationally efficient general purpose pruned 1-bit look-ahead SDM realizations: the Efficient Trellis SDM and the Pruned Tree SDM.
- Investigation of the possibilities to add a prediction cost function to the look-ahead filter for improving the lossless data compression gain. Derivation, implementation, and performance evaluation of the Pruned Tree SDM for SA-CD.
- Comparison of all the major look-ahead sigma-delta modulation techniques, resulting in a clear motivation for selecting a specific look-ahead algorithm depending on the performance requirements.
- The increased stability of a look-ahead SDM enables the use of very aggressive noise-shaping filters. Analysis demonstrated a limit on the achievable SNR of a 1-bit modulator. A design approach is derived for realizing the maximum possible SNR, and a method is introduced for calculating the loop-filter corner frequency that results in the optimal noise shaping.

List of publications

Papers

D. Reefman and E. Janssen. Enhanced Sigma Delta structures for Super Audio CD application. In *Proceedings of the AES 112'th convention*, 2002. preprint 5616, May 10-13 Munich

D. Reefman, E. Janssen, J. D. Reiss, and M. B. Sandler. Improved compression of DSD for Super Audio CD. In *Proceedings of the AES 112th convention*, 2002. preprint 5617 May 10-13 Munich

D. Reefman and E. Janssen. DC analysis of high order sigma delta modulators. In *Proceedings of the AES 113th convention*, 2002. preprint 5693 October 5-8 Los Angeles

D. Reefman and E. Janssen. *Signal processing for Direct Stream Digital*. Philips IP&S, Available upon request, December 2002

P. Harpe, D. Reefman, and E. Janssen. Efficient Trellis-type Sigma-Delta Modulator. In *Proceedings of the AES 114'th convention*, 2003. Preprint 5845, March 22-25 Amsterdam

E. Janssen and D. Reefman. Super-audio CD: an introduction. In *IEEE Signal Processing Magazine*, volume 20, July 2003. Invited paper

D. Reefman, J. D. Reiss, E. Janssen, and M. B. Sandler. Stability analysis of limit cycles in high order sigma delta modulators. In *Proceedings of the AES 115th convention*, 2003. preprint 5936 October 10-13 New York

E. Janssen and D. Reefman. Advances in Trellis based SDM structures. In *Proceedings of the AES 115'th convention*, 2003. Preprint 5993, October 10-13 New York

E. Knapen, D. Reefman, E. Janssen, and F. Bruekers. Lossless compression of one-bit audio. *Journal of the Audio Engineering Society*,

52(3):190–199, March 2004. Invited paper

D. Reefman and E. Janssen. One-bit Audio: An Overview. *Journal of the Audio Engineering Society*, 52(3):166–189, March 2004. Invited paper

R. M. Aarts, D. Reefman, and E. Janssen. Super audio CD - An overview. In *Proceedings of the 18th International Congress on Acoustics (ICA2004)*, volume 3, pages 2179–2182, April 2004. Paper No.We4.E.3, Kyoto, Japan. Invited paper

E. Janssen, E. Knapen, D. Reefman, and F. Bruekers. Lossless compression of one-bit audio. In *Acoustics, Speech, and Signal Processing (ICASSP2004)*, 2004. May 17-41 Montreal. Invited paper

D. Reefman, J. D. Reiss, E. Janssen, and M. B. Sandler. Description of limit cycles in feedback Sigma Delta Modulators. In *Proceedings of the AES 117th convention*, 2004. preprint 6280 October 28-31 San Francisco

E. Janssen and D. Reefman. DSD compression for recent ultra high quality 1-bit coders. In *Proceedings of the AES 118th convention*, 2005. preprint 6470 May 28-31 Barcelona

D. Reefman, J. D. Reiss, E. Janssen, and M. B. Sandler. Description of limit cycles in Sigma Delta Modulators. *IEEE Transactions on Circuits and Systems I*, 52:1211–1223, June 2005

E. Janssen and A. H. M. v. Roermund. A performance comparison of look-ahead sigma-delta modulation algorithms. *IEEE Transactions on Circuits and Systems I*, 2010. Submitted

Patents

D. Reefman, P. Harpe, and E. Janssen. Noise-shaping device and method with improved lossless compression and good audio quality for high fidelity audio. U.S. Patent Application 2007/0290906 Al. filed July 15, 2004

E. Janssen and D. Reefman. Sigma-delta modulator with a quantizer/gain element. U.S. Patent 7,330,142. filed Apr 13, 2004, issued Feb 12, 2008

E. Janssen and D. Reefman. Adaptive filtering. U.S. Patent Application 2007/0052556 A1. filed Apr 15, 2004

E. Janssen and D. Reefman. Generating bit-streams with higher compression gains. U.S. Patent 7,218,263 B2. filed October 8, 2004, issued May 15, 2007

Summary

In this thesis the possibilities of look-ahead sigma-delta modulation are investigated, with the objective to improve the signal conversion quality compared to that of a normal SDM. This investigation is first made for look-ahead sigma-delta modulation in general, while from chapter 6 onwards the focus is on general purpose 1-bit DD conversion. In chapter 10 a look-ahead realization that is specifically optimized for Super Audio CD usage is presented.

In chapter 2, the basics of traditional sigma-delta modulation are reviewed, and a number of SDM performance indicators are introduced. These indicators can be divided in two classes, i.e. indicators that are of a general nature and that are applicable to all data converter types, and indicators that are specific for Sigma-Delta Modulators.

In chapter 3, an investigation is made of the potential difference between the SINAD performance of a (non-linear) 1-bit SDM for a steady-state signal and that of a non-steady-state signal. For this purpose a time-domain SINAD measurement method is introduced that allows for SINAD performance characterization of non-steady-state signals. The outcome of this investigation is that there is no significant difference between the performance of non-steady-state and steady-state signals, and that the classical frequency domain SINAD measurement suffices.

An abstract model of a noise-shaping quantizer is derived in chapter 4. This model introduces the concept of a cost function, which is an indication of the quality of a signal or an encoding solution. In the case of a normal SDM, the output of the loop filter can be considered as a cost value. The negative feed-back strategy attempts to minimize the absolute value of this cost, but can not guarantee this result because of the loop delay.

In chapter 5, the noise-shaping quantizer model is modified to support look-ahead, and the main look-ahead principle is explained. Instead of

relying on a feed-back strategy that attempts to minimize the instantaneous cost value, i.e. a local optimization strategy that does not attempt to find the solution that is optimal in a global sense, the impact of the selected output symbol on the future cost values is taken into account by a quantizer with look-ahead. Although only with an infinite amount of look-ahead the global optimal encoding solution can be found, it is concluded that already with a limited amount of look-ahead an improvement in the signal conversion performance can be obtained. The main disadvantage of incorporating look-ahead into an SDM is the increase of the resource costs. The benefits, primarily an increase of the stability and a reduction of the distortion, are the biggest in the case of a 1-bit converter, since here the quantizer is severely non-linear.

Up till now, the analysis was performed for look-ahead Sigma-Delta converters in general. The possibilities for realizing a look-ahead enabled Sigma-Delta ADC are investigated, but because of the large number of identical loop filters that are required, the idea of an ADC with look-ahead is rejected. The potential for a DD converter with look-ahead is large, but also here the computational cost should be reduced compared to the straight-forward full look-ahead solution. From this point onwards, the work focusses on 1-bit look-ahead DD sigma-delta modulation, although most of the results can be directly applied to multi-bit look-ahead sigma-delta modulation.

The possibilities for reducing the computational cost of a look-ahead DD converter are explored in chapter 6. Two possibilities are identified, i.e. an optimization of the full look-ahead algorithm, and a reduction of the solution space. It is found that the possibilities for optimizing the full look-ahead algorithm are limited, and that the full look-ahead approach can only by used for look-ahead depths up to 16-20 samples. The reduction of the solution space, a process called pruning, offers good possibilities for the realization of a large look-ahead depth at a limited computational cost. Several ideas for realizing pruned look-ahead modulators are presented, that are explored in detail in the next chapters.

Chapter 7 presents a full analysis of the Trellis sigma-delta modulation algorithm. This algorithm is the first pruned look-ahead sigma-delta modulation algorithm found in literature. It is a derivative of the full look-ahead algorithm and uses concepts from Trellis (Viterbi) decoding, hence the name. In the Trellis sigma-delta modulation algorithm, at all times, a total number of 2^N potential solutions (paths) are investigated, of which the most recent N symbols are different for all the solutions. The output symbol of the converter is found by tracing back any of the 2^N paths L time steps. This approach results in a converter that has

an improved linearity and better stability than a normal SDM, although the computational load is very high.

In chapter 8, an efficient derivative of the Trellis SDM, called the Efficient Trellis SDM, is introduced. Instead of exploring 2^N solutions in parallel, only M solutions out of the possible 2^N are tracked. This is possible since only a fraction of all the 2^N solutions contributes to the final output. The selection of which paths to keep is based on the accumulated path cost, i.e. paths with a low cost have a large probability to be part of the output and are selected, whereas more expensive paths are rejected. Compared to the Trellis sigma-delta modulation algorithm, the computational load is reduced by several orders of magnitude, while at the same time improvements in the linearity and stability are obtained.

A final reduction of the computational load is achieved in chapter 9, where the Pruned Tree sigma-delta modulation algorithm is presented. This algorithm is a practical realization of the pruned look-ahead approach as derived in chapter 6. A total of M paths are tracked, with no constraints on the solution space coverage imposed. The result of this approach is that a performance level that is slightly better than that of the Efficient Trellis sigma-delta modulation algorithm is realized, at a significantly reduced computational load.

In chapter 10, the Pruned Tree sigma-delta modulation algorithm for SA-CD is presented. The algorithm has a better compatibility with Super Audio CD, because it generates bitstreams that result in a high lossless data compression gain. This is is achieved by adding a cost function to the look-ahead filter that measures the predictability of the output signal. This addition results in an output bitstream that is of a high signal quality, but that is also very predictable, such that the amount of required disc storage space after lossless data compression reduces. This reduction of the required storage space is of great importance, since it solves potential playback duration issues. The addition of the prediction cost function has only a minimal impact on the SNR, while the distortion and the noise-modulation performance of the converter are strongly reduced, resulting in the ideal Sigma-Delta converter for high-end audio applications.

All of the previously discussed look-ahead techniques are compared in chapter 11. The outcome of this comparison is that the normal Pruned Tree sigma-delta modulation algorithm is the best choice if the converter is not intended for audio applications, since it offers the highest SNR, very good linearity, and the largest stability at the minimal computational cost. In the case of a high-end application for Super Audio CD, the Pruned Tree sigma-delta modulation algorithm for SA-CD is the

best choice because of the constant in-band noise-floor and the higher compression gains that are obtained on the output bitstream.

In chapter 12, an investigation is made of the apparent limit on the obtainable SNR of a 1-bit look-ahead SDM. The outcome of this investigation is that there is a point of maximal noise shaping, which depends on the filter order. At the point of maximal noise shaping the system is critically stable, and increasing the corner frequency of the loop filter above this point will not change the noise-shaping characteristics, i.e. the look-ahead system forces the same noise shaping as obtained for the critically stable point. If, instead of increasing the loop-filter corner frequency further, a higher filter order is selected in combination with a lower loop-filter corner frequency, a more aggressive noise shaping can be realized that results in a higher SNR. Since the more aggressive noise shaping causes a reduction of the stability of the SDM, more parallel paths are required to stabilize the system. Although with this approach a world-record SNR for a 1-bit noise-shaped signal has been achieved, it is still far away from the limits imposed by information theory. As such, in practice the SNR is only limited by the amount of available computational power that is required to stabilize the higher order filters.

Finally, in chapter 13 the general conclusions on the work described in this thesis are presented.

Samenvatting

In dit proefschrift worden de mogelijkheden van *look-ahead* sigma-deltamodulatie onderzocht, met als doel de kwaliteit van de signaalconversie te verbeteren ten opzichte van die van een normale SDM. Dit onderzoek wordt eerst gedaan voor sigma-deltamodulatie in het algemeen, terwijl vanaf hoofdstuk 6 de focus gericht is op 1-bit volledig digitale SDM conversie voor algemeen gebruik. In hoofdstuk 10 wordt een realisatie besproken die voor specifieke toepassing voor Super Audio CD geoptimaliseerd is.

In hoofdstuk 2 wordt de basis van klassieke sigma-deltamodulatie besproken en worden een aantal SDM-kwaliteitsindicatoren geïntroduceerd. Deze indicatoren zijn onder te verdelen in twee groepen, dat wil zeggen indicatoren die algemeen van aard zijn en die van toepassing zijn op alle soorten signaalomzetters en indicatoren die specifiek zijn voor sigma-deltaomzetters.

In hoofdstuk 3 wordt onderzocht of er bij een (niet lineaire) 1-bit SDM een verschil bestaat tussen de SINAD voor niet-stationaire en voor stationaire signalen. Met dit doel wordt er een tijdsdomein-SINAD-meetmethode geïntroduceerd die het mogelijk maakt om een SINAD-karakterisering te doen voor niet-stationaire signalen. De uitkomst van dit onderzoek is dat er geen significant verschil bestaat tussen de SINAD voor niet-stationaire en stationaire signalen en dat de traditionele SINAD-methode in het frequentiedomein voldoet.

Een abstract model van een *noise-shaping*-kwantisator wordt afgeleid in hoofdstuk 4. Dit model introduceert het idee van een kostenfunctie om aan te geven wat de kwaliteit van een signaal of een coderingsoplossing is. In het geval van een normale SDM kan de uitgang van het lusfilter gezien worden als de waarde van een kostenfunctie. De negatieve terugkoppeling probeert de absolute waarde van deze waarde te minimaliseren, maar kan niet garanderen dat dit lukt vanwege de vertraging in de lus.

In hoofdstuk 5 wordt het *noise-shaping*-kwantisatormodel aangepast om met *look-ahead* overweg te kunnen en wordt het hoofdidee van *look-ahead* uitgelegd. In plaats van op de terugkoppelingsstrategie te vertrouwen, die probeert om de instantane kosten te minimaliseren, oftewel een lokale optimalisatiestrategie die niet probeert om het globale optimum te vinden, wordt de invloed van het gekozen uitgangssymbool op de toekomstige kosten geëvalueerd door een kwantisator met *look-ahead*. Alhoewel alleen met een oneindig grote mate van *look-ahead* de globaal-optimale coderingsoplossing gevonden kan worden, wordt geconcludeerd dat al met een beperkte mate van *look-ahead* een verbetering in de signaalomzettingskwaliteit kan worden bereikt. Het voornaamste nadeel van het toepassen van *look-ahead* in een SDM is de toename van de benodigde rekenkracht. De voordelen, voornamelijk een toename van de stabiliteit en een afname van de vervorming, zijn het grootst in het geval van een 1-bit omzetter omdat hier de kwantisator erg niet-lineair is.

Tot dusver werd de analyse uitgevoerd voor *look-ahead* sigma-deltaomzetters in het algemeen. De mogelijkheden voor het realiseren van een analoog naar digitaal sigma-deltaomzetter met *look-ahead* worden onderzocht, maar vanwege het grote aantal identieke lusfilters dat nodig is wordt het idee van een analoog-naar-digitaalomzetter met *look-ahead* verworpen. Het potentieel van een digitaal-naar-digitaalomzetter met *look-ahead* is groot, maar ook in dit geval moet de benodigde rekenkracht verminderd worden ten opzicht van de rechttoe-rechtaan volledige *look-ahead* oplossing. Vanaf dit punt is de focus van het werk gericht op 1-bit *look-ahead* digitaal-naar-digitaal sigma-deltamodulatie, alhoewel de meeste resultaten direct toegepast kunnen worden op meer-bits *look-ahead* sigma-deltamodulatie.

De mogelijkheden voor het verminderen van de benodigde hoeveelheid rekenkracht voor digitaal-naar-digitaalomzetting met *look-ahead* worden bekeken in hoofdstuk 6. Twee mogelijkheden worden geïdentificeerd: een optimalisatie van het *full look-ahead*-algoritme en een verkleining van de oplossingsruimte. Er wordt geconcludeerd dat de mogelijkheden voor het optimaliseren van het *full look-ahead*-algoritme beperkt zijn en dat de *full look-ahead*-methode slechts gebruikt kan worden voor een *look-ahead*-diepte van 16-20 samples. Het verkleinen van de oplossingsruimte, een proces dat *pruning* genoemd wordt, biedt goede mogelijkheden voor het realiseren van een grote *look-ahead*-diepte met een beperkte hoeveelheid rekenkracht. Een aantal mogelijkheden voor het realiseren van een *pruned look-ahead*-modulator worden gepresenteerd. Deze ideeën worden uitgewerkt in de volgende hoofdstukken.

In hoofdstuk 7 wordt een volledige analyse van het *Trellis*-sigma-delta-

modulatiealgoritme gepresenteerd. Dit algoritme is het eerste *pruned look-ahead*-sigma-deltamodulatiealgoritme dat in de literatuur beschreven is. Het is een afgeleide van het *full look-ahead*-algoritme en maakt gebruik van ideeën uit de *Trellis-* (*Viterbi-*) decodering. Dit verklaart tevens de naam van het algoritme. In het *Trellis*-sigma-deltamodulatiealgoritme wordt, altijd, een totaal van 2^N mogelijke oplossingen (paden) onderzocht, waarvan de meest-recente N symbolen verschillen tussen alle oplossingen. Het uitgangssymbool van de omzetter wordt bepaald door een willekeurig pad van de 2^N mogelijke paden L tijdstappen terug te volgen. Deze aanpak resulteert in een omzetter met een betere lineariteit en grotere stabiliteit dan een normale SDM. De benodigde rekenkracht is echter zeer groot.

In hoofdstuk 8 wordt een efficiënte afgeleide van de *Trellis*-sigma-deltamodulator, *Efficient Trellis SDM* genoemd, geïntroduceerd. In plaats van 2^N oplossingen tegelijk te bekijken worden er slechts M van de 2^N mogelijke oplossingen bijgehouden. Dit is mogelijk omdat slechts een deel van alle 2^N oplossingen bijdraagt aan het uiteindelijke conversieresultaat. De keus welke paden te behouden is gebaseerd op de geaccumuleerde padkosten. Met andere woorden, paden met een lage kostenwaarde hebben een grotere kans om bij te dragen aan het conversieresultaat en worden behouden terwijl dure paden verworpen worden. In vergelijking met het *Trellis*-sigma-deltamodulatiealgoritme is de benodigde rekenkracht verminderd met enkele orde groottes terwijl tegelijkertijd de lineariteit en stabiliteit verbeterd is.

Een laatste vermindering van de benodigde hoeveelheid rekenkracht wordt gerealiseerd in hoofdstuk 9, dat het *Pruned Tree*-sigma-deltamodulatiealgoritme beschrijft. Dit algoritme is een praktische realisatie van de *pruned look-ahead*-aanpak zoals die in hoofdstuk 6 afgeleid is. Er worden in totaal M paden bijgehouden, zonder enige beperking op de oplossingsruimtedekking. Het resultaat van deze aanpak is een kwaliteitsniveau dat iets beter is dan van het *Efficient Trellis*-sigma-deltamodulatiealgoritme, terwijl de benodigde hoeveelheid rekenkracht significant verminderd is.

In hoofdstuk 10 wordt het *Pruned Tree*-sigma-deltaalgoritme voor SACD gepresenteerd. Dit algoritme heeft een betere compatibiliteit met Super Audio CD omdat het bitstromen genereert die resulteren in een hoge compressieratio. Dit wordt gerealiseerd door een kostenfunctie toe te voegen aan het *look-ahead*-filter wat de voorspelbaarheid van het uitgangssignaal meet. Deze toevoeging zorgt voor een bitstroom met een hoge signaalkwaliteit die tegelijkertijd erg voorspelbaar is, waardoor de benodigde hoeveelheid opslagruimte na toepassing van de verliesvrije

datacompressie afneemt. Deze afname van de benodigde hoeveelheid opslagruimte is van groot belang omdat dit problemen met betrekking tot potentiële speelduur oplost. Het toevoegen van een kostenfunctie die de voorspelbaarheid meet heeft slechts een minimale impact op de SNR, terwijl het zorgt voor een sterk verminderde signaalvervorming en ruismodulatie, waardoor de ideale sigma-deltaomzetter voor high-end audiotoepassingen ontstaat.

Alle eerder besproken *look-ahead*-technieken worden vergeleken in hoofdstuk 11. De uitkomst van deze vergelijking is dat het normale *Pruned Tree*-sigma-deltamodulatiealgoritme de beste keus is als de omzetter niet gebruikt wordt voor audiotoepassingen, daar deze de hoogste SNR, een goede lineariteit en de grootste stabiliteit realiseert met de minste rekenkracht. In het geval van een high-end toepassing voor Super Audio CD is het *Pruned Tree*-sigma-deltamodulatiealgoritme voor SA-CD de beste keus vanwege de constante ruisvloer en de hogere compressieverhouding die behaald wordt op de gegenereerde bitstroom.

In hoofdstuk 12 wordt de ogenschijnlijke limiet op de haalbare SNR van een 1-bit *look-ahead*-SDM onderzocht. De uitkomst van dit onderzoek is dat er een punt bestaat waarop maximale *noise shaping* gerealiseerd wordt en dat dit punt een functie van de filterorde is. Op het punt van de maximale *noise shaping* is het systeem kritiek stabiel en het verhogen van de kantelfrequentie van het lusfilter tot boven dit punt zorgt niet meer voor een verandering in de *noise-shaping*-curve, wat wil zeggen dat het *look-ahead*-systeem dezelfde *noise shaping* forceert als op het kritiek stabiele punt. Als in plaats van een hogere kantelfrequentie van het lusfilter een hogere orde lusfilter wordt gekozen in combinatie met een lagere kantelfrequentie, dan is het mogelijk om een agressievere *noise shaping* te realiseren wat zorgt voor een hogere SNR. Omdat de agressievere *noise shaping* een vermindering van de stabiliteit veroorzaakt zijn er meer parallelle paden nodig om het systeem te stabiliseren. Alhoewel met deze aanpak een wereldrecord gezet is voor de SNR van een 1-bit omzetter, is deze SNR nog ver verwijderd van de grens die afgedwongen wordt door de wetten van de informatietheorie. In de praktijk is de SNR daarom alleen gelimiteerd door de hoeveelheid rekenkracht die beschikbaar is om hoge-orde lusfilters te stabiliseren.

Als laatste worden in hoofdstuk 13 de generieke conclusies over het in dit proefschrift beschreven werk gepresenteerd.

Dankwoord

Het schrijven van een proefschrift in deeltijd valt niet mee. Sterker nog, ik vond het af en toe echt afzien om 's avonds na een dag werken opnieuw aan het werk te moeten gaan. De steun, interesse en continue support van een aantal mensen heeft me geholpen om door te zetten en dit werk tot een succesvol einde te brengen. Deze mensen wil ik graag bedanken.

Allereerst wil ik mijn baas, Leo Warmerdam, eerst bij Philips Research en tegenwoordig bij NXP Semiconductors, bedanken voor de mogelijkheid om een dag per week te besteden aan het schrijven van deze thesis. Zonder deze support had ik dit werk niet tot een goed einde kunnen brengen.

Vervolgens wil ik mijn promotor, prof.dr.ir. Arthur van Roermund, bedanken voor de prettige en stimulerende discussies die we samen hadden. Het plannen van een afspraak met je viel niet altijd mee vanwege je volle agenda, maar als we eenmaal in gesprek waren dan vergat je meestal spontaan dat er nog iemand anders was die ook een afspraak met je had. Bedankt dat je deze passie voor techniek met me wilde delen.

Dit dankwoord wil ik ook gebruiken om mijn collega's bij NXP te bedanken voor de prettige werkomgeving. Specifiek wil ik Kostas Doris en Claudio Nani bedanken voor de discussies over look-ahead gerelateerde zaken. Arnoud van der Wel, Berry Buter, Robert Rutten, Henk Jan Bergveld, Athon Zanikopoulos en Alessandro Muroni voor het regelmatig informeren naar de status van mijn promotie. Robert van Veldhoven voor het delen van ervaringen en frustraties bij het schrijven van een thesis. Thijs Withaar (Trident) wil ik bedanken voor het lezen van mijn proefschrift en voor het begrijpen dat ik bijna nooit meer tijd had om als vrienden iets te gaan doen. Derk Reefman, mijn mentor toen ik begon bij Philips Research, wil ik bedanken voor de fijne samenwerking in de SA-CD tijd. Tevens bedankt voor het aanmoedigen om een promotie te gaan doen en voor de feedback op eerdere versies van dit manuscript.

Mijn schoonouders wil ik bedanken voor het altijd beschikbaar zijn en het regelmatig, maar niet te vaak, vragen of het boek al af is. Jan, bedankt voor het samen klussen. Dan zijn er natuurlijk nog mijn ouders die me gemaakt hebben tot wie ik ben. Bedankt voor alles wat jullie me geleerd en meegegeven hebben. Bedankt voor de steun en de belangstelling in mijn vorderingen tijdens het promotietraject. Inderdaad pa, van hard werken ga je niet dood, maar je wordt er wel moe van.

Als laatste en meest belangrijke wil ik mijn vrouw Rachel bedanken. Bedankt voor je continue steun gedurende deze lange drie jaar en het niet aflatende vertrouwen in een goede afloop. Bedankt voor de twee schatten van kinderen die je me hebt gegeven. Collin en Amy bedankt dat jullie papa 's nachts lieten slapen en het begrip dat jullie toonden als papa thuis was maar geen tijd voor jullie had omdat hij moest werken. Zonder de steun van jullie drieën was het schrijven van dit proefschrift onmogelijk geweest. Bedankt dat jullie er zijn.

Biography

Erwin Janssen was born in Ede, The Netherlands in 1976. He received the M.Sc degree *cum laude* in electrical engineering, with additional degree in computer science, from the University of Twente, Enschede, The Netherlands, in 2001. He joined the Mixed-Signal Circuits and Systems group of Philips Research Laboratories, Eindhoven, The Netherlands, in 2001, to work for several years on various aspects of signal processing for super audio compact disk. In 2006 he joined NXP Semiconductors, Eindhoven, The Netherlands, and since then he has specialized in mixed-signal IC design, with a special focus on calibration techniques for high speed data conversion systems. Other research interests include digital signal processing and sigma-delta modulation. In 2007 he started, next to his position at NXP Semiconductors, working toward the Ph.D. degree in electrical engineering at the Eindhoven University of Technology, The Netherlands, of which the results are presented in this dissertation.