Low power Decimation Filter for 16 bit Audio $\Delta\Sigma$ modulator & Efficient Method for Calculating Feedback DAC Errors for Digital Correction in $\Delta\Sigma$ A/D converters

A Project Report

submitted by

A. SANTOSH KUMAR REDDY

in partial fulfilment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS

May 2011

CERTIFICATE

This is to certify that the thesis titled **Low power Decimation Filter for 16 bit Audio ΔΣmodulator & Efficient Method for Calculating Feedback DAC Errors for Digital Correction in ΔΣ A/D converter**, submitted by **A.Santosh kumar reddy**, to the Indian Institute of Technology Madras, for the award of the degree of **Master of Technology**, is a bona fide record of the work done by him under my supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Dr. Nagendra Krishnapura
Advisor,
Associate Professor,
Dept. of Electrical Engineering,
IIT-Madras, 600 036

Place: Chennai
Date:

ACKNOWLEDGEMENTS

First of all, I would like to express deepest sense of gratitude to my advisor Dr. Nagendra Krishnapura whose energy, enthusiasm and constant support inspired me from the beginning till the end of the project. I am indebted to him for his ideas, valuable guidance and the encouragement throughout.

I would like to thank various faculty members of IIT Madras from whom I have benefited as a student. I especially thank Dr. Shanthi Pavan for his lectures on VLSI Data Conversion Circuits motivated me to work in Analog and Mixed-Signal design.

I express my sincere gratitude to Shankar and Kunal whose support had proved invaluable during the course of the project. I would not have been able to finish in time if not for their help with design tools and the concepts of decimation filter. I would like to thank Sreenu for his support and discussions during the project. I would also like to thank lab supporting staff Mrs. Sumathy and Mrs. Janaki for their help and support.

Last but not least, I would like to thank Ankesh, Anil, Sravan and Sandeep, my friends in the DSDL lab for their support and the friendly atmosphere in lab. I would also like to thank my classmates who made my life in IIT memorable experience.

Finally, I dedicate this thesis to my parents and brother who make my any little success meaningful.

ABSTRACT

This project involves the design of a Low power Decimation filter for a 16bit audio $\Delta\Sigma$ Analog to Digital converter with a sampling frequency of 1.536 MHz. A multistage, multirate approach is used for decimator structure to reduce the order of the filters. Hogenuaer structure is used for SINC section which reduces power consumption. Polyphase decomposition for halfband filter section reduces power consumption approximately by half. The design is implemented in 0.18 μm CMOS process from UMC. It occupies an area of 490 μm $\times$ 483 μm. The design consumes a total power of about 52.5 μW (mostly dynamic power) from a 1.8 V supply.

Other part of this work involves to implement a algorithm for acquiring and correcting the errors of the feedback DAC used in a multibit $\Delta\Sigma$ ADC. This method requires simple computation, does not add excess loop delay, and requires no reconfiguration of the $\Delta\Sigma$ modulator. The example design is implemented for $OSR = 32, OBG = 3, 16$ level DAC in 0.18 μm CMOS process from UMC. It occupies a core area of 493 μm $\times$ 478 μm with power consumption of 185 μW from a 1.8 V. Once the errors are determined and the correction values stored, this design is inactive after which the power consumption will be 11.5 μW in RAM block.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABBREVIATIONS

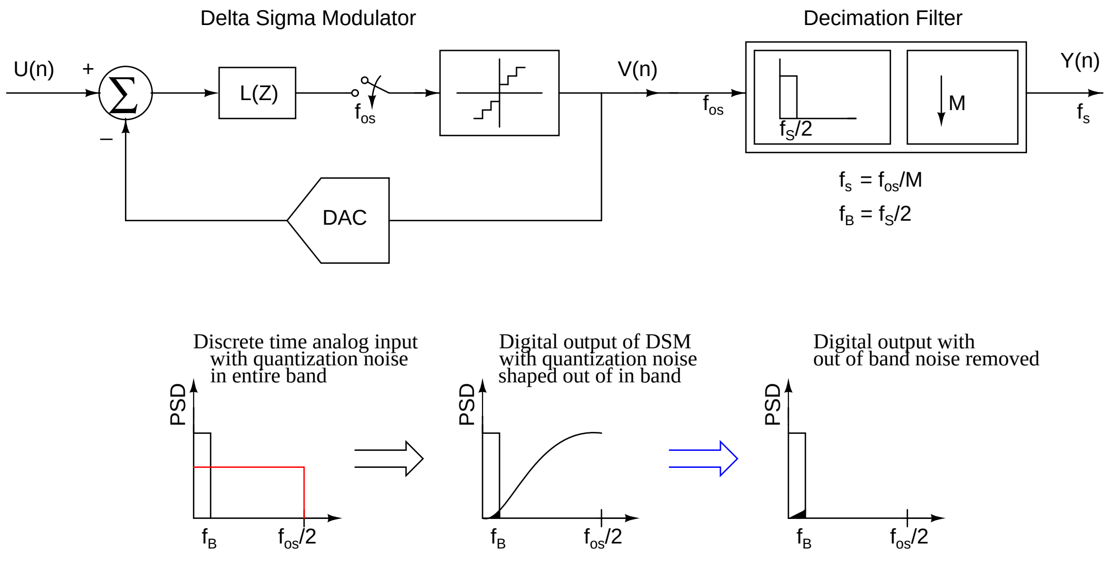
ADC	Analog to Digital Converter
DAC	Digital to Analog Converter
PSD	Power Spectral Density
LTI	Linear Time Invariant
NTF	Noise Transfer Function
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
CIC	Cascaded Integrator Comb
LPF	Low pass filter
CSD	Canonical Signed Digits
VCD	Value Change Dump
DEM	Dynamic Element Matching
DWA	Data Weighted Averaging
OBG	Out of Band Gain
LSB	Least Significant Bit
MSA	Maximum Stable Amplitude
SDM	Sigma Delta Modulator
SNR	Signal to Noise Ratio
SQNR	Signal to Quantization Noise Ratio

CHAPTER 1

Introduction

1.1 Motivation

Sigma delta data converter which are is widely applied in areas related to multimedia, communication systems, industrial applications etc. High fidelity audio applications are always in need of high dynamic range data converters. Sigma delta converters are capable of providing high dynamic range, high reliability at reduced chip cost.



Delta Sigma Modulator
Decimation Filter
U(n)
+
−
L(Z)
f_{os}
V(n)
f_{os}
$f_S/2$
M
Y(n)
f_s
DAC
$f_s = f_{os}/M$
$f_B = f_S/2$
Discrete time analog input with quantization noise in entire band
Digital output of DSM with quantization noise shaped out of in band
Digital output with out of band noise removed
PSD
PSD
PSD
f_B
$f_{os}/2$
f_B
$f_{os}/2$
f_B
$f_{os}/2$

Figure 1.1: Block diagram of decimation filter for a $\Delta\Sigma$ modulator. f_B is the maximum inband frequency. f_{os} = Oversampling frequency.

The noise-shaping technique of oversampled $\Delta\Sigma$ A/D converter shown in Figure 1.1. Once the quantization noise is shaped out of the passband, a low pass filter(LPF) is required to remove the out of band components and a decimator is needed to the bring the sampling rate back to the Nyquist rate from the oversampled rate. Both these tasks are performed by a decimation filter, which as its name suggests, low pass filters and decimates (brings down the sampling rate) the input signal.

The aim of this project is to design a low power decimation filter for audio range $\Delta\Sigma$ modulator. It becomes clear that multistage, multirate decimator structure will leads to low power design. Since the low pass filtering and the decimation are performed on the output signal of the $\Delta\Sigma$ modulator, the entire decimation filter system works in the digital domain.

1.2 Organization

Chapter 2 introduces decimation filter requirements and specifications of the $\Delta\Sigma$ modulator and decimation filter are considered. A multistage design architecture is derived for decimation filter.

Chapter 3 gives the details of the circuit implementations of the three stages of the decimation filter. Architectures for both SINC and halfband filter sections are explained.

Chapter 4 concludes the decimation filter design with simulation results after synthesis, placement and routing.

Chapter 5 explains DAC nonlinearity and a digital correction procedure to remove DAC mismatches. A effecient way to determine the feedback DAC errors is discussed.

Chapter 6 explains the implementation and hardware details for digital correction algorithm.

Chapter 7 concludes the digital correction design and thesis with simulation results after synthesis, placement and routing.

CHAPTER 2

Decimation Filter Design

Decimation Filter is one of the key components of a *Sigma Delta* Modulator that converts low resolution digital signal into a high resolution digital signal at a lower sampling rate. High resolution signal is obtained by *averaging* the modulator output samples. This chapter deals with the design of decimation filter. Implementation aspects of decimation filter are discussed in next chapter.

2.1 Decimation Filter Requirements

There are many application where the signal at given sampling rate needs to be converted to another signal with different sampling rate. The key elements in sampling rate conversion are *up-sampler* and *Down-sampler*. The process of reducing sampling rate by just dropping samples is called *Down-sampling*, the time domain equation of which is given in (2.1)

$$y[n] = x[nM] \tag{2.1}$$

Where M is down-sampling factor. Equation (2.1) means that output $y[n]$ is one out of every M samples of $x[n]$ or every M^{th} sample of $x[n]$. Sampling in time domain results in frequency translation of input signal spectrum to multiples of sampling frequency. Because of this frequency translation, the spectral components at $Kf_s \pm f_c$ (where K=1,2,...$\frac{M}{2}$) are aliased to baseband. This aliasing effect for discrete time wideband signal like output of a $\Delta\Sigma$ modulator is explained in Figure (2.1). So before down-sampling process, spectral components at alias bands should be attenuated. "*Hence, the digital LPF used in decimator must attenuate these aliasing components around Kf_s to the level much lower than the noise floor in baseband*".

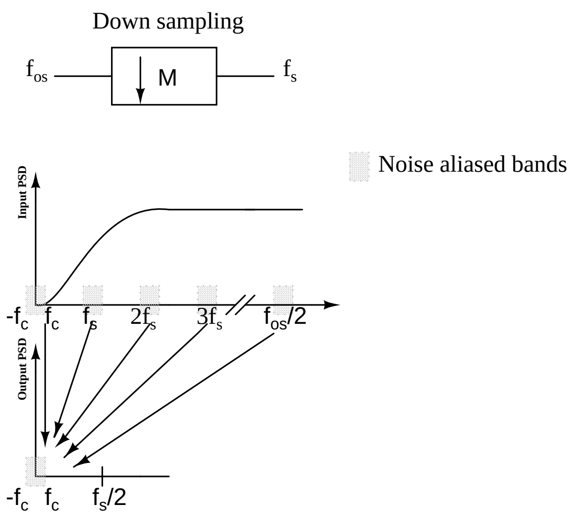


Figure 2.1: Down-sampling a discrete-time sine wave by a factor, $M = 2$

From the system level decimation filter should

- *Suppress the out-of-band quantization noise*
- *Resample the over-sampled signal to lower sampling rate*

2.2 Cascade equivalace of decimation filters

A complex *multirate system* is formed by an interconnection of the up-sampler, the down-sampler, and components of an LTI digital filter. In most applications, these two appear in cascade as shown in Figure 2.2. For efficient computational results a particularly useful cascade equivalence property for our design is shown in Figure 2.2 and the equivalence is proved by the manipulation of frequency domain properties of the signals.

2.3 Decimation filter specifications

The specifications of the decimation filter are dependant on the overall specification from the sigma-delta converter. A $\Delta\Sigma$ modulator for audio range applications having *16*bit dynamic range has chosen for decimation filter. This decimation filter design is targeted for low power as the modulator[1]. The specifications of

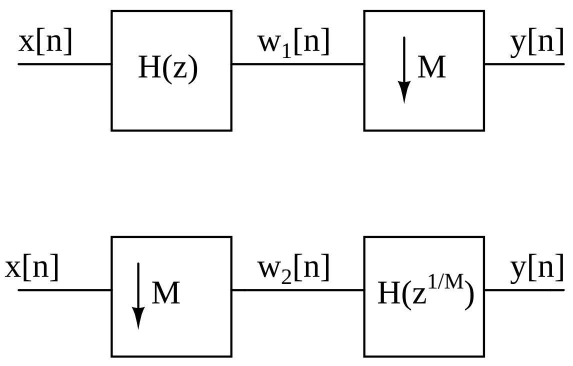


Figure 2.2: cascade equivalence of decimation filter

Table 2.1: Targeted ΔΣ modulator specifications

Sampling Rate(f_{os})	1.536 MHz
OSR	32
Modulator order	3
Out-of-band gain	3
Dynamic range	96 dB(16bit)

the targeted sigma-modulated are summarised in Table 2.1. The noise-shaping of the modulator is determined by the Noise Transfer Function or NTF of the ΔΣ modulator. The NTF of the ΔΣ modulator is given in (2.2) and the magnitude response of this NTF is plotted in Figure 2.3.

$$NTF(z) = \frac{(z-1)(z^2-1.994z+1)}{(z-0.3556)(z^2-0.6603z+1)} \tag{2.2}$$

The shape of the magnitude response of NTF will explains *noise shaping* phenomenon of ΔΣ modulator. This high pass filtering for quantization error will shift the inband quantization noise to out of band and thus the decimation to nyquist rate should be accompanied by low pass filtering which removes the out of band noise components. Since the input bandwidth for audio range applications is 24 KHz, we essentially need a structure with a low pass filter with cutoff frequency 24 KHz and a decimator with decimating factor 32. The output of the decimation filter will be a digital signal with sampling rate 1.536 MHz/32 =48 KHz which is the nyquist rate.

A typical spectrum of the output from this ΔΣ modulator for input signal

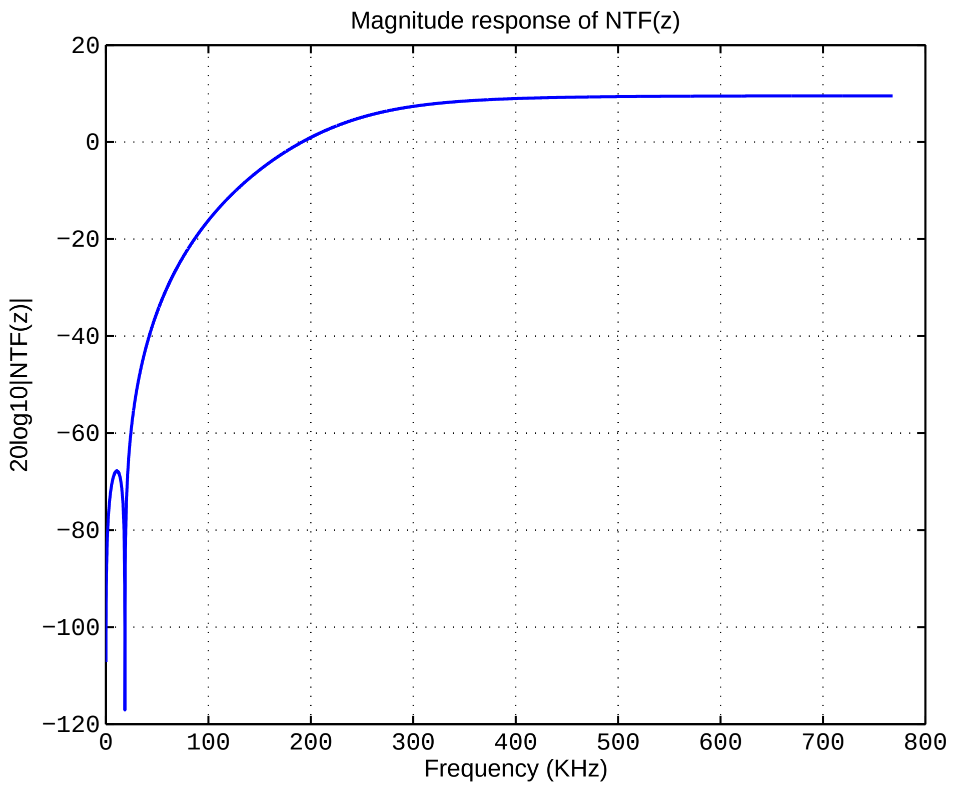


Figure 2.3: Magnitude response of NTF of the ΔΣ modulator

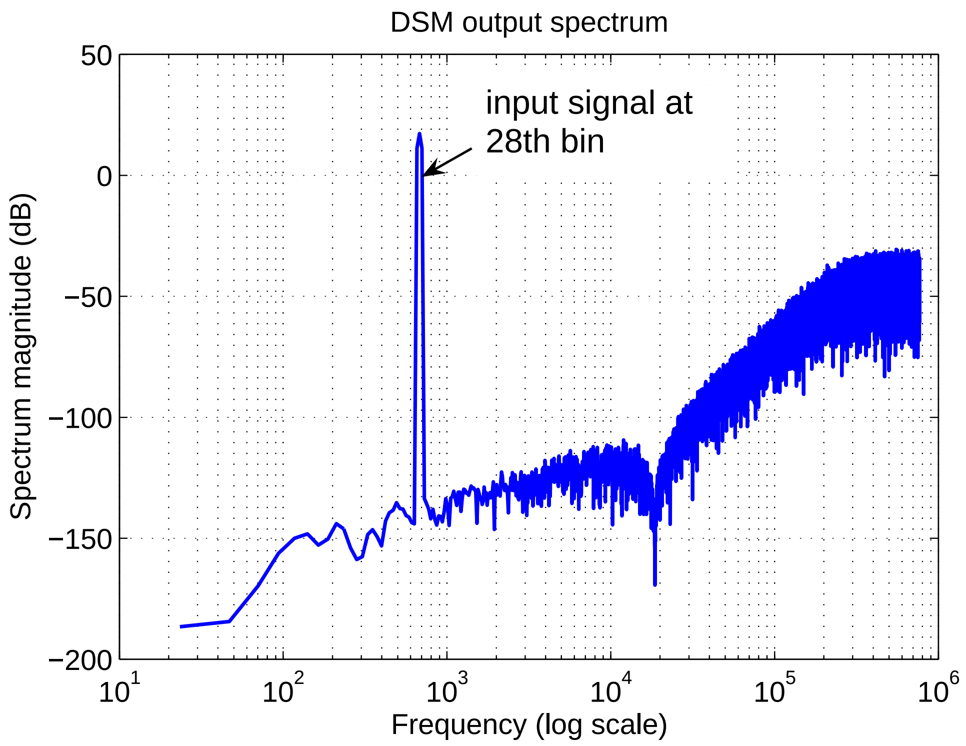


Figure 2.4: Output spectrum of the ΔΣ modulator

Table 2.2: Decimation filter specifications

Input No.of bits	4
Input sampling rate	1.536 MHz
Passband edge	21.6 KHz
Max passband ripple (r_p)	$\leq$ ($\pm$ 0.05 dB)
Dynamic range	96 dB(16bit)
Output frequency	48 KHZ

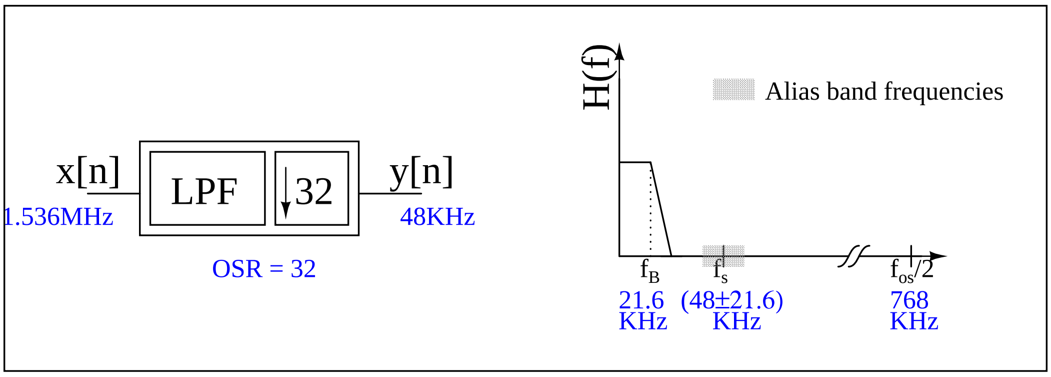
$Asin(2\pi(f_{in}/f_S)n)$ is shown in Figure 2.4. A is Maximum Stable Amplitude or MSA of the modulator. For the simulation, $\Delta\Sigma$ modulator having OSR 32 and 16-level quantizer is modeled in MATLAB using Schreier's Delta Sigma Toolbox[2]. A 2^{16}-point FFT point is taken at the output of $\Delta\Sigma$ modulator spectrum having input signal at 28^{th} bin. Figure 2.4 explains noise shaping of quantization noise with slope of +60 dB for 3^{rd} order $\Delta\Sigma$ modulator.

From the knowledge of $\Delta\Sigma$ modulator specifications, the decimation filter average the 4-bit stream to improve the ADC resolution. The specifications of the decimation filter are summarised in Table 2.2.

2.4 Decimation filter design

The simplest design approach for decimation filter is single stage filter shown in Figure 2.5. In this a digital LPF is designed to meet the decimation filter specifications and it is followed by a down-sampler. Since the signal is oversampled by a large factor, the ratio of signal bandwidth(f_B) to half the oversampled rate, relative bandwidth, is very less. Filtering at oversampled rate and then down sampling to the nyquist rate requires a very sharp filter working at the oversampled rate, which explains the Figure 2.5. Therefore the required order of the filter is very high.

A simple Parks-McClellan optimal FIR filter order estimation[3] is used to find the required filter order. The MATLAB commands given in Figure 2.5 will find the approximate order of the filter that meet the input specifications. For the specifications given in Table 2.2 required order of the filter $N \approx 1238$. Here passband ripple(r_p) is 0.05 dB and stopband ripple(r_s) is 90 dB, which will satisfy the 16 bit dynamic range condition.



N = firpmord(f_{os},a,dev)

a = magnitudes at passband and stopband = [1 0]

dev = [(10^(r_p/20)-1)/(10^(r_s/20)-1) 10^(-r_s/20)]

Figure 2.5: Single stage design of decimation filter and MATLAB command to estimate the order of the filter for single stage.

Power dissipation of digital filters is directly propotional to thier frequency of operation. In single stage design entire digital LPF operates at oversampling rate, increases the power dissipation. The disadvantages of single stage design approach are

- *Excessively large filter orders because of narrow transition bandwidths.*
- *High power dissipation as the entire filer operates at f_{os}.*
- *Inefficient use of hardawre is because one out of every M computed samples is passed to output.*

2.4.1 Multi-stage decimation

The disadvantages ot single stage can be overcome by multistage design. Filtering the alias band noise in multiple stages and down sampling by an appropriate factor in each stage ensures effecient decimation. The decimator is designed as a cascade of two or more stages such that the overall decimation ratio is product of the decimation ratio of each of the stage. This is called multistage decimation.

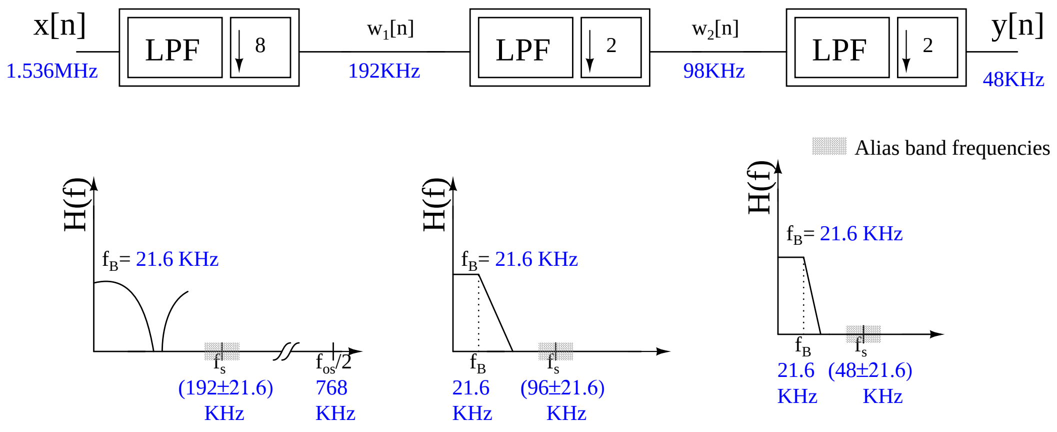


Figure 2.6: Three stage design of decimation filter

Advantages of multistage design can be better explained with the Figure 2.6. Here decimation filter designed in three stages. Total decimation factor 32 divided as 8x2x2. As we can see from Figure 2.6, lower order filters are required in each stage because of relatively wider transition bands. Since the first stage is operating at oversampling freqecy, proper slection of filter is needed at this stage. The simplest low-pass FIR filter with no multiplier is a filter with all its coefficients equal to unity, which is a Moving Average (MA) filter. Using moving average filter, commonly known as SINC filter, three stage design of decimation filter is shown in Figure 2.7. The design issues of each section in decimator chain are explained in following sections.

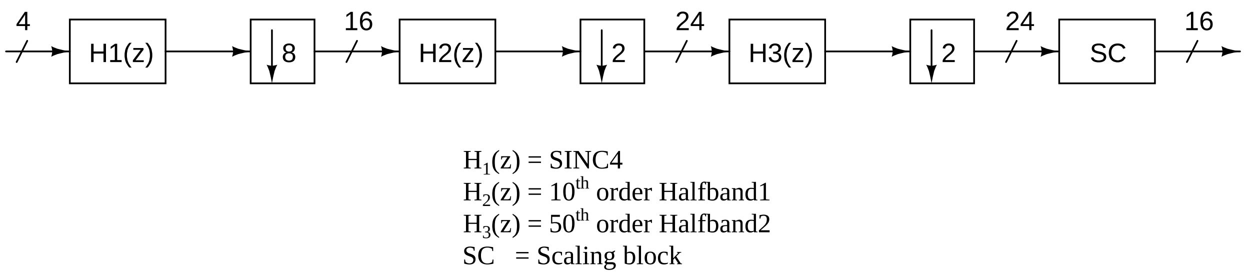


Figure 2.7: Decimation filter chain

2.5 SINC4 filter characteristics

A CIC filter is a special class of linear phase, finite impulse response (FIR) filter. CIC filters do not require multipliers and use a limited amount of storage.

Therefore, CIC filters are more efficient than conventional FIR filters, especially in fixed-point applications. Furthermore, it can be used to decimate the data by large factor, allowing easier implementation of following stages. The input-output relation of CIC filter with M number samples for averaging is given in 2.3.

$$y[n] = \frac{1}{M}\left(x[n] + x[n-1] + x[n-2] + ... + x[n-M+1]\right) \tag{2.3}$$

Applying transform

$$H(z) = \frac{1}{M}(1 + z^{-1} + z^{-2} + ... + z^{-(M-1)} \quad = \frac{1}{M}\Sigma_{n=0}^{M-1} z^{-n} = \frac{1}{M}\left(\frac{1-z^{-M}}{1-z^{-1}}\right) \tag{2.4}$$

Frequency response of MA filter is given by

$$H(e^{jw}) = \frac{1}{M}\left(\frac{1-e^{-jMw}}{1-e^{-jw}}\right) \quad = \frac{1}{M}\left(\frac{sin(\frac{Mw}{2})}{sin(\frac{w}{2})}\right) e^{-jw(\frac{M-1}{2})}, \text{where} \quad w = \frac{2\pi f}{f_{os}} \tag{2.5}$$

The magnitude response is

$$|H(e^{jw})| = \frac{1}{M}\left|\frac{sin(\frac{Mw}{2})}{sin(\frac{w}{2})}\right| \tag{2.6}$$

Magnitude response of the MA filter shown in Figure 2.8. As it's magnitude response is *sinc* function, the MA filter is called *sinc* filter. The magnitude response has nulls at $\frac{Kf_{os}}{M} = Kf_s$. As explained in Figure 2.1 for decimating the signal from f_{os} to f_s, the regions around Kf_s should be attenuated to avoid aliasing to baseband. With the nulls at Kf_s positions, SINC filter can attenuate the spectral components around Kf_s.

As explained in section 1.3, for a third order $\Delta\Sigma$ modulator quantization noise spectrum has roll off that goes as f^3 or +60 dB. In order to attenuate the rising shpaed quantization noise, the order of the SINC filter should be atleast one order higher than that of the $\Delta\Sigma$ modulator. A $SINC^4$, cascade of four SINC filters has a roll of that goes as $\frac{1}{f^4}$ will suffuciently attenuate the aliasband quantization

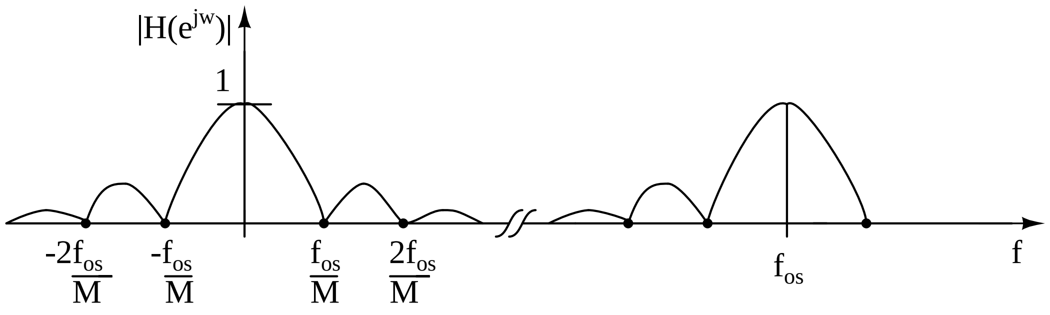


Figure 2.8: Manitude response of MA filter

noise.

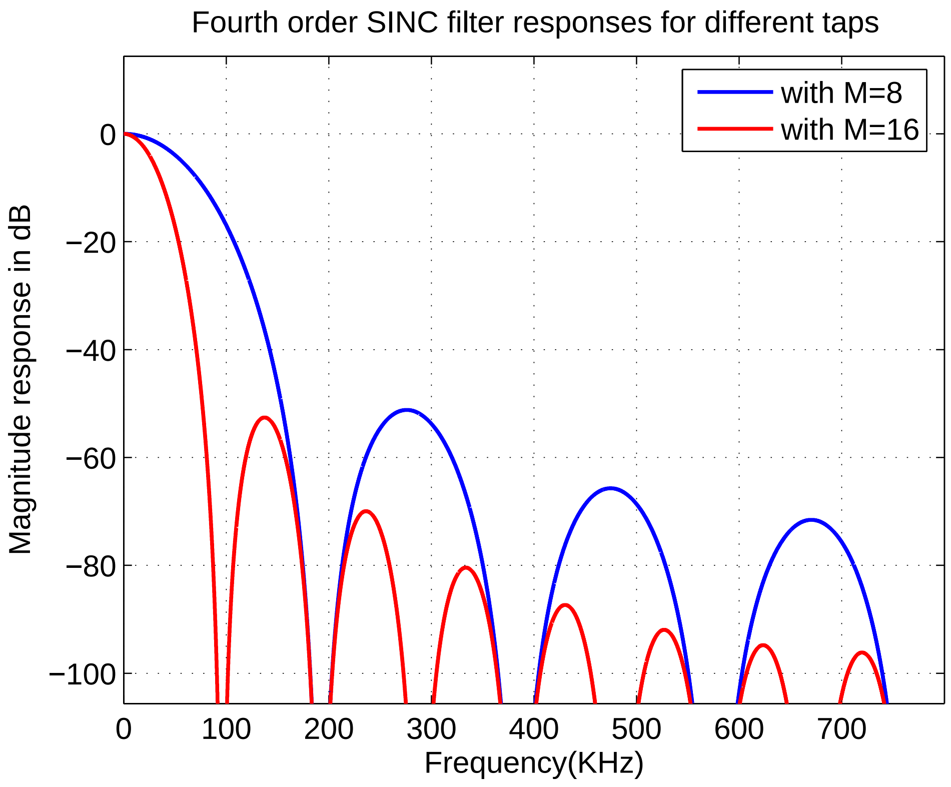


Figure 2.9: Magnitde response of fourth order SINC filter for M=8 and M=16

The frequency response of fourth order SINC filter for M=8 and 16 is shown in Figure 2.9. The variation of maximum passband droop value and worst alias rejection for different M values is summarized in Table 2.3. As the number of samples for averaging M increases, the worst alias rejection become better but the passband droop will increase. Usually a maximum passband droop of 3 dB or above is difficult to correct. So a fourth order 8 tap SINC filter is used at the first stage of decimation filter. Refer Figure 2.7.

Table 2.3: Variation of maximum droop and worst alias rejection for different M values

Number of samples for averaging	**Passband droop(in dB) at 21.6 KHz**	**Worst alias rejection(in dB) at 170.4(192-21.6) KHz**
M = 8	-0.7149	-71.83
M = 16	-2.931	-74.07
M = 32	-12.45	-83.54

2.6 Halfband filter characteristics

Halfband filters are a class of equiripple FIR filters, where the transition region is centered at one quarter of the sampling rate, or fs/4. Specifically, the end of the passband and the beginning of the stopband are equally spaced on either side of fs/4. In other words filter exibits a symmetry with respect to halfband frequecy fs/4 or $\frac{\pi}{2}$. An important property of the halfband filter is that about 50% of coefficents are zero. This reduces the number of multiplications required in its implementation. The order of the filter N_{ord} = 4P+2, where P ε 1,2,3 ....

- $\frac{N_{ord}}{2}$-1 coefficients are zero.
- Middle coefficient is always 0.5.
- Remaining $\frac{N_{ord}}{2}$+1 coeffients are symmetric about the middle coefficient

Halfband filter have their 6 dB frequency at 0.25*f_S. Hence maximum downsampling that is allowd after filtering is two. Halfband filters are implemented as a polyphase structure that includes inherent downsampling[4]. Implementation details are explained in next chapter.

2.6.1 Halfband one

The input for the halfband filter comes from the $SINC^4$ fitler at 192 KHz. When downdampled by a factor of two, the noise in the region 74.4 KHz to 96 KHz alias to in band. Halfband filters designed with passband edge at 21.6 KHz. A tenth order filter sufficiently attenuates the aliasing noise and the passband ripple of the filter meets specifications. The frequency response of halfband one filter is shown in Figure 2.10.

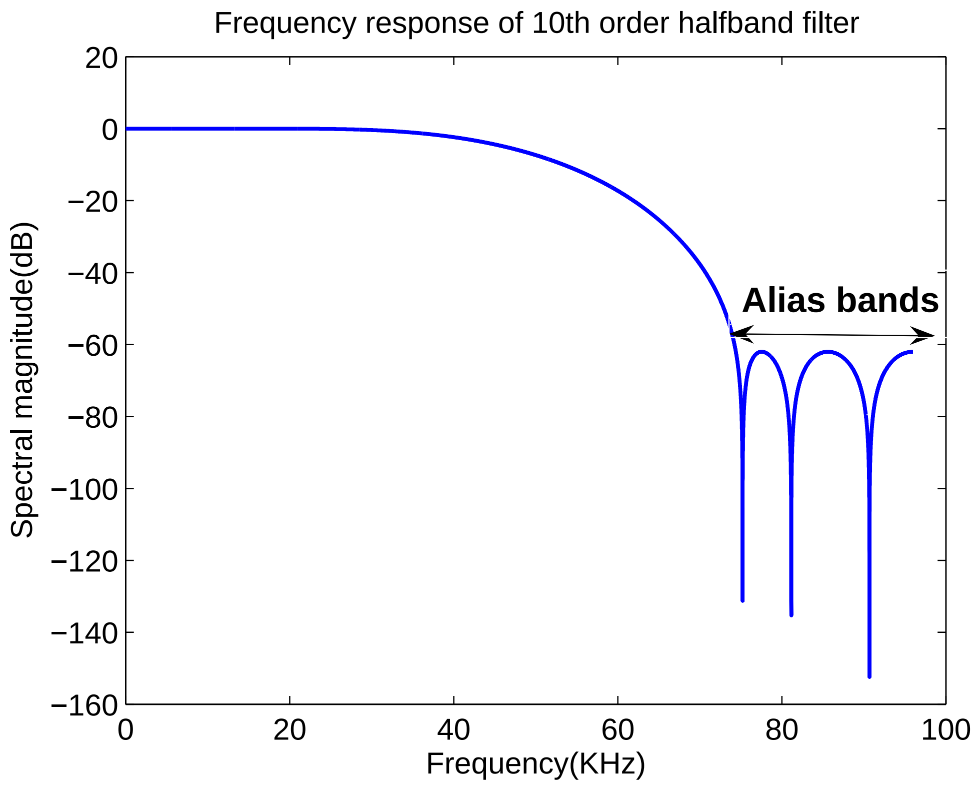


Figure 2.10: Frequresponse of halfband one filter

2.6.2 Halfband two

This second halfband filter provides the final filtering before the signal is downsampled to he nuquist rate. The input for the second halfband filter comes from the first halfband filter at 96 KHz. Noise in the region 26.4 KHz to 48 KHz alias to in band when downsampled by a factor of two. Here the transition band is less. Higher order filters give sharper transition band, lesser ripples and consume more power. Passband edge is fixed at 21.6 KHz. To get the sharper transition band and for required passband ripple a 50^{th} order halfband filter is required. The frequency response of the halfband two filter is shown in Figure 2.11. The totlal response of the three stage decimation filter shown in Figure 2.12. Filter has pass band droop of -0.7519 $\dot{d}$B at 21.6 KHz and worst alias rejection of -50.4 dB at 26.4 KHz.

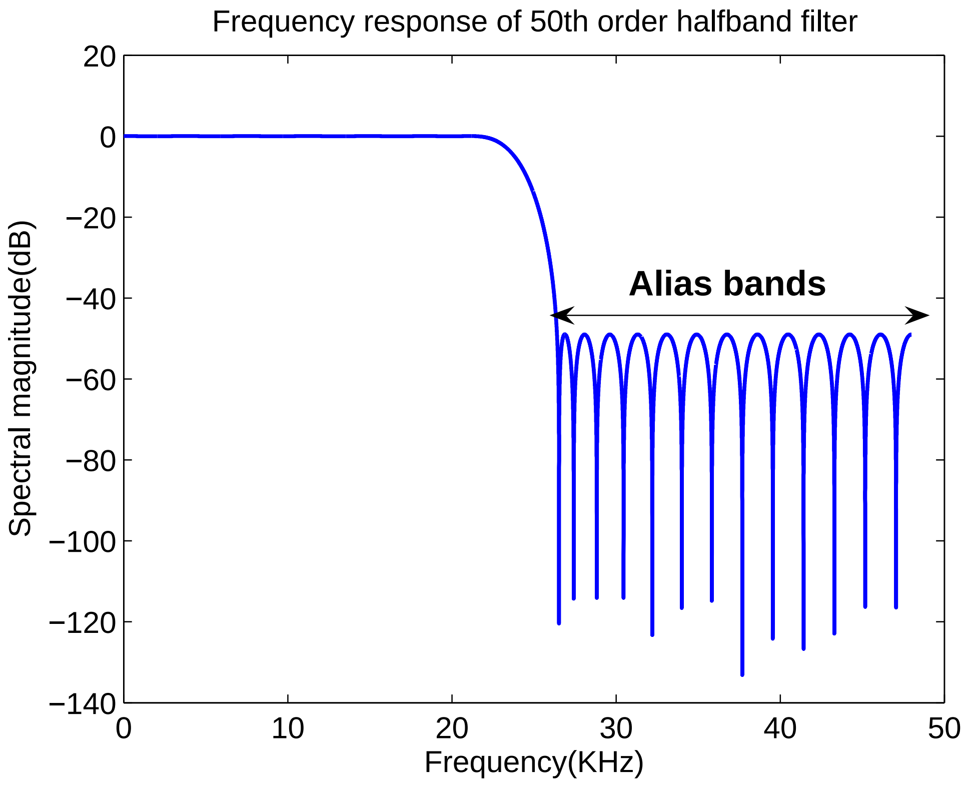


Figure 2.11: Frequresponse ofbhalfband two filter

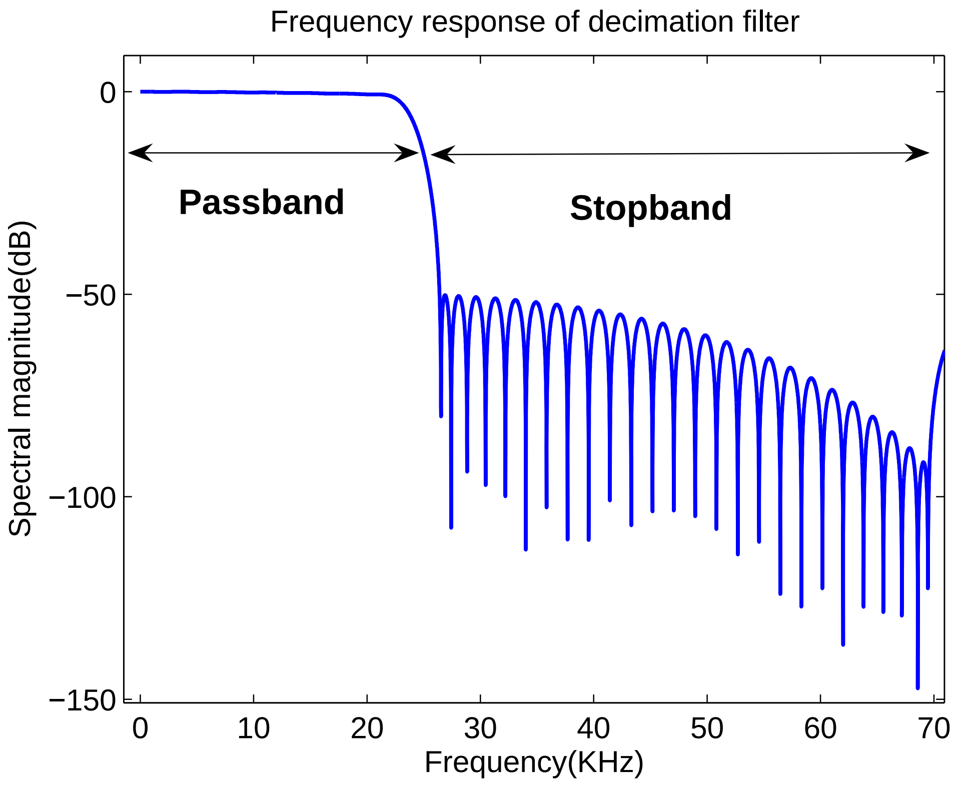


Figure 2.12: Total frequency response of the three stage decimation filter

The output response of the decimation filter for single tone input at 2.625 KHz is shown in Figure 2.13.

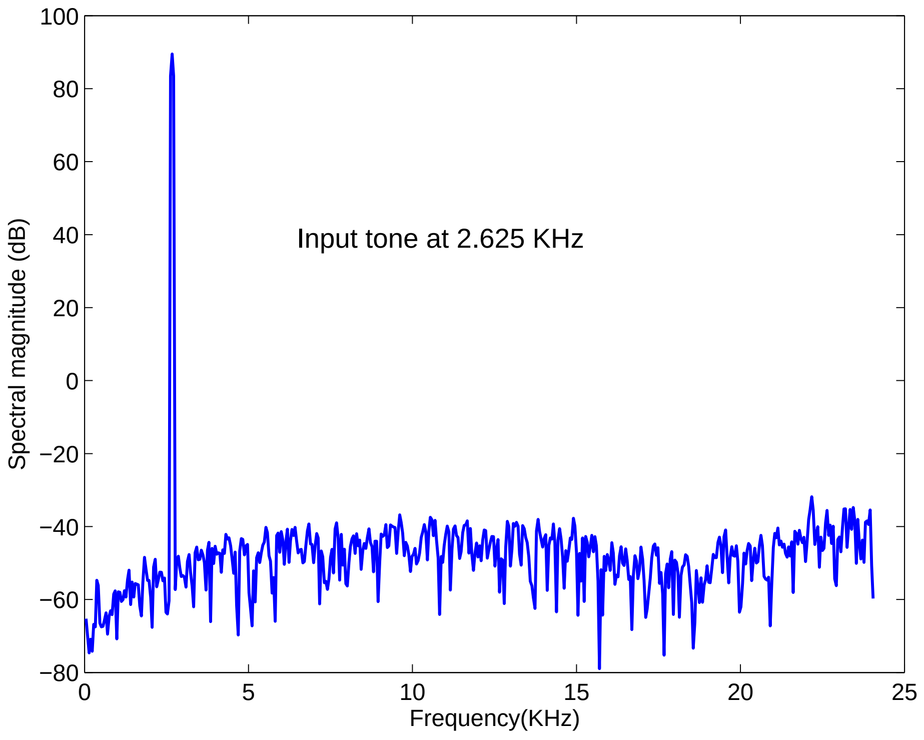


Figure 2.13: Output response of the decimation filter

CHAPTER 3

Architecture

The conventional architecture used to implement the since filter in the first stage of a decimation filter is the Cascade Integrator Comb(CIC) or Hogenauer structure. Hogenauer devised a flexible, multiplier free CIC filter that can handle large sampling rate changes suitable for hardware implementation.

3.1 Cascade Integrator Comb (CIC) filter

The transfer function of a 8 tap fourth order SINC filter is,

$$H(z) = \left(\frac{1 - z^{-8}}{1 - z^{-1}} \right)^4 \tag{3.1}$$

From the cascade equivalence explained in section 2.2, equation 3.1 can be decomposed into accumulator(or integrators) and differentiators(or comb filter). This process explained in Figure 3.1. So, CIC filter is an infinite impulse response (IIR) filter followed by a finite impulse response (FIR) filter. This classic Hogenauer[5] structure for fourth order SINC filter shown in Figure 3.2.

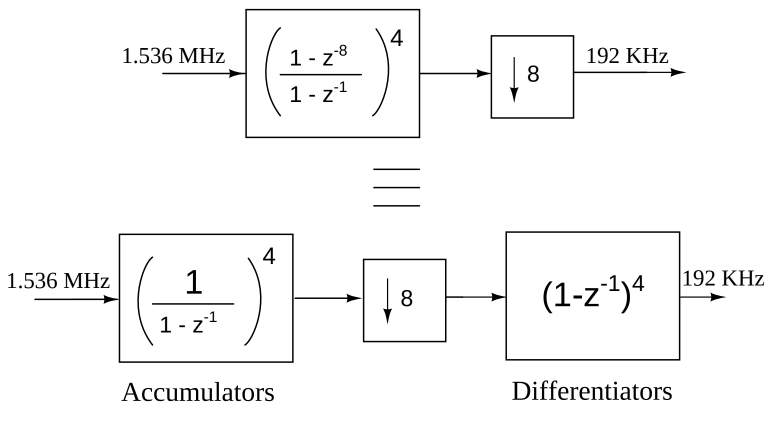
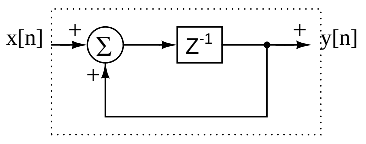
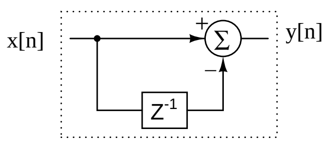


Figure 3.1: Decomposition fo sinc decimation filter which leads to CIC implementation



(a) Accumulator, $\frac{Y(z)}{X(z)} = \frac{z^{-1}}{(1-z^{-1})}$



(b) Differentiator, $\frac{Y(z)}{X(z)} = 1-z^{-1}$

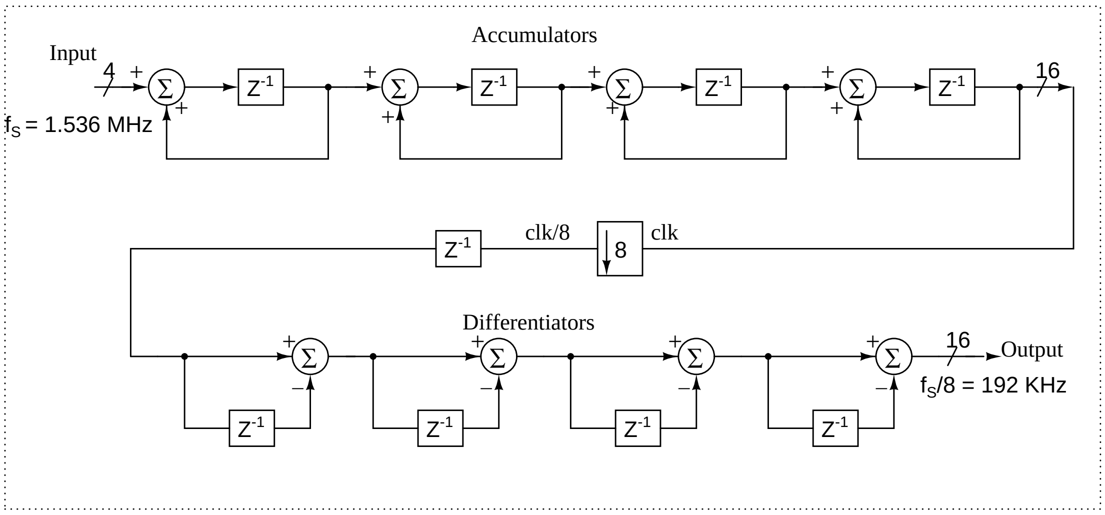


Figure 3.2: The classic Hogenauer structure with cascade of accumulators and differentiators

In this fourth order CIC filter, the integrator section consists of a cascade of four digital integrators operating at oversampling rate f_{os}, and the comb section consists of a cascade of four differentiators with digital delay of 'M', operates at lower sampling rate $\frac{f_{os}}{M} = 192\,\text{KHz}$. This reduces both hardware and power.

There is inherent downsampling in the first differentiator. But deliberately a register is inserted at the output of the fourth accumulator. This pipelining register avoids the high speed switching data to propagate through the combinational adders of the differentiator. Similarly accumulators are implemented in a retimed(register in the forward path instead of the feedback path) fashion. Both pipelining and retiming structures helped in reducing power by 40%.

The accumulator adds the previous state value with its current input and can overflow. To avoid overflow the minimum width of all the registers in the CIC filter is

$$B_{width} = B_{in} + Klog_2^N = 16 \qquad (3.2)$$

- B_{in} = Number of input bits = 4
- K = Number of SINC in cascade = 4
- N = Number of samples taken for averaging = 8

The sigma delta output swings on either side of its DC value. But its output is coded in binary which represents numbers form 0 to 15. The DC offset, here 7.5, can't be represented as a digital word. Next stage filters are operate with negative coefficients, hence to represent CIC filter output in two's complement arithmetic form, the DC offset is to be subtracted from 7.5* DC gain of SIN^4 = 7.5 * 8^4.

3.2 Polyphase implementation of halfband Filters

In half-band filters, the number of taps is reduced by 50% since the odd coefficients are zeros. Refer section. This considerably reduces the hardware and the power consumpiton. As a result, we use half-band FIR filters instead of direct conventional FIR filter. The half-band FIR filter is designed using the Remez algorithm, which gives the filter coefficients. The filter coefficients for half-band one filter are tabulated in Table 3.1.

The poly-phase implementation exploits the symmetry of coefficients of the FIR filters. This reduces the computational complexity since only half the number of coefficients is used to implement the decimation filter. This poly-phase implementation reduces the power consumption by approximately half compared to the conventional structure because, the filter operates at decimated rate 96 KHz, instead of the input rate 192 kHz. The poly-phase implementation of half-band one FIR filter is shown in Figure 3.3.

The transfer function of the 10^{th} order halfband one filter obtained in Table can be written as equation where h_k, k = 0,1..10 are the coefficients.

$$H_f(z) = \sum_{k=0}^{k=10} h_k H z^{-k} \tag{3.3}$$

$$= h_0 + h_2 z^{-2} + h_4 z^{-4} + h_6 z^{-6} + h_8 z^{-8} + h_{10} z^{-10} + h_5 z^{-5} \tag{3.4}$$

$$= E_0(z^2) + E_1(z^2) \tag{3.5}$$

where

$$E_0(z) = h_0 + h_2 z^{-1} + h_4 z^{-2} + h_6 z^{-3} + h_8 z^{-4} + h_{10} z^{-5} \tag{3.6}$$

$$= h_0(1 + z^{-5}) + h_2(z^{-1} + z^{-4}) + h_4(z^{-2} + z^{-3}) \tag{3.7}$$

$$E_1(z) = h_5 z^{-2} \tag{3.8}$$

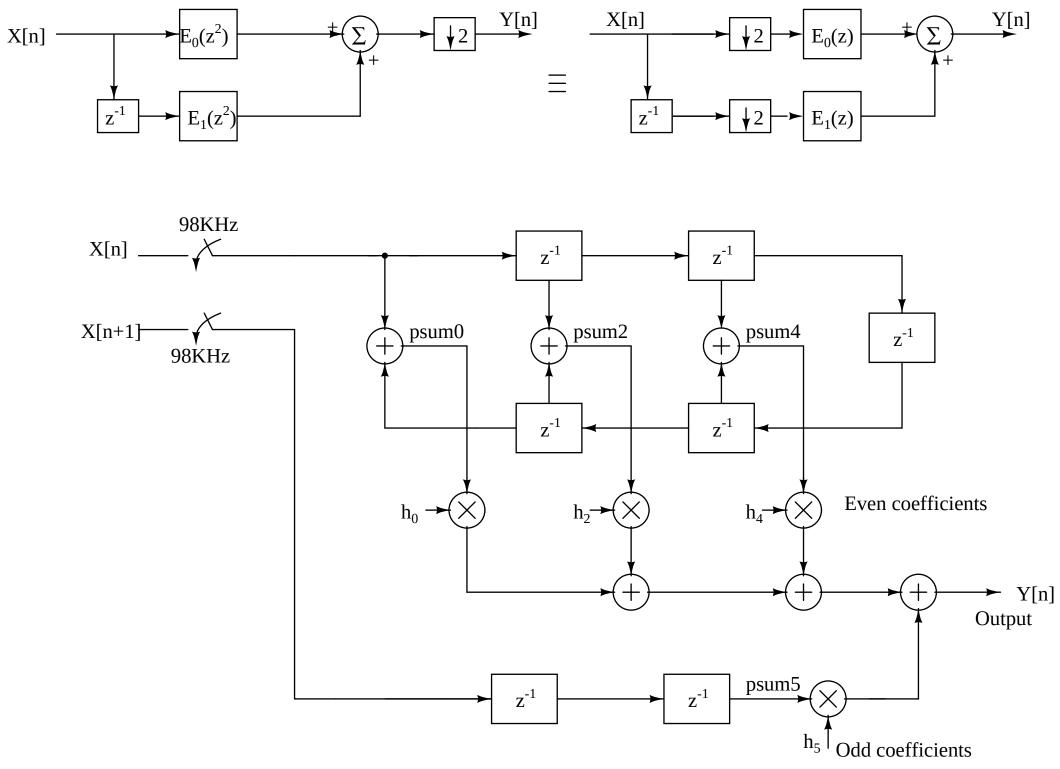


Figure 3.3: Polyphase implementation of halfband one filter

Coefficients are encoded in Canonical Signed Digits (CSD) that reduces the number of computations[4] . This is similar to the Booth multiplication. For

Table 3.1: Halfband one filter coefficients. Only 4 filter coefficients are shown because of the symmetry $h_k = h_{10-k}$, $k = 0, 1, \dots 10$

coefficients h_k	**Value of h_k**	**Canonical Signed Digits (CSD) form**
h_0	0.0112	$2^{-6} - 2^{-8} - 2^{-10}$
h_2	-0.0608	$-2^{-4} + 2^{-9} - 2^{-10}$
h_4	0.3	$2^{-2} + 2^{-4} - 2^{-6} + 2^{-8} - 2^{-10}$
h_5	0.5	2^{-1}

Example,

$$0.9375 = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-4} \tag{3.9}$$

$$= 2^{0} - 2^{-4} \quad (CSDform) \tag{3.10}$$

The frequency response of the Halfband filter one is shown in Figure. The stop

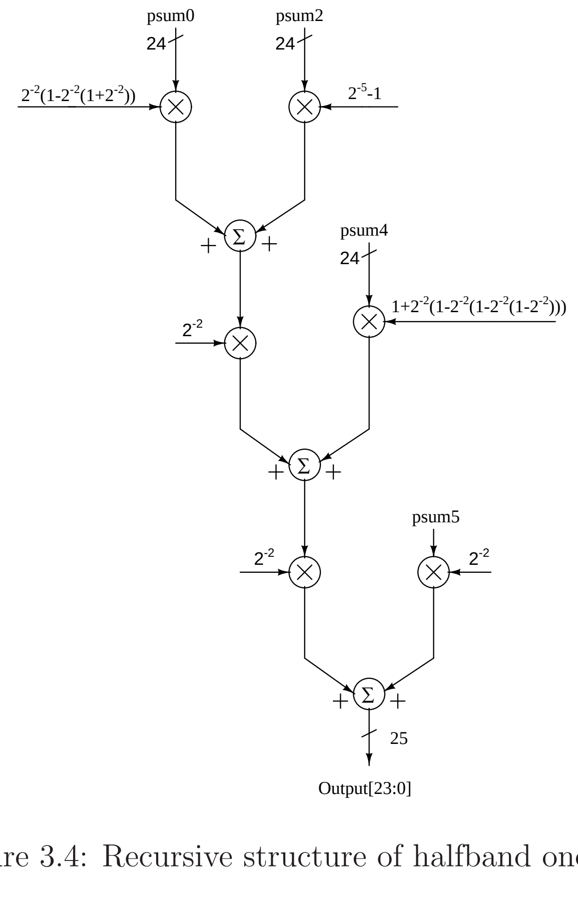


Figure 3.4: Recursive structure of halfband one filter

band attenuation is around 60 dB below the in band noise floor, the internal states of the filter have to be represented with 10 extra bits than the signals in band resolution(16 bit). The number of bits chosen was 24. Hence 48 dB is the maximum out off band attenuation with respect to the in band noise floor.

Coefficient multiplication with 2^{-1} means, right shifting the input register which introduces truncation error. So the coefficient multiplication in each delay element is split recursively. This ensures that round-off error that occurs is minimized. This recursive splitting for coefficient multiplication is shown in Figure 3.4.

The second halfband filter provides sharp filtering, requires higher orders. The filter coefficients for half-band two filter are tabulated in Table 3.2. The polyphase structure of halfband two filter operates at decimated rate 48 KHz, reduces the required power. CSD form and recursive splitting is used to decrease the truncation error.

Table 3.2: Halfband two filter coefficients. Only 13 filter coefficients are shown because of the symmetry $h_k = h_{50-k}$, $k = 0, 1, \ldots 50$

coefficient h_k	**Value of h_k**	**Canonical Signed Digits (CSD) form**
h_0	0.0029	$2^{-8} - 2^{-10}$
h_2	-0.003	$-2^{-8} + 2^{-10}$
h_4	0.0044	$2^{-8} + 2^{-11}$
h_6	-0.0063	$-2^{-7} + 2^{-9} - 2^{-11}$
h_8	0.0087	$2^{-7} + 2^{-10}$
h_{10}	-0.0119	$-2^{-6} + 2^{-8}$
h_{12}	0.0159	2^{-6}
h_{14}	-0.0214	$-2^{-5} + 2^{-7} + 2^{-9} + 2^{-12}$
h_{16}	0.0289	$2^{-5} - 2^{-9} - 2^{-11}$
h_{18}	-0.0403	$-2^{-5} - 2^{-7} - 2^{-10} - 2^{-12}$
h_{20}	0.0509	$2^{-4} - 2^{-8} + 2^{-10}$
h_{22}	-0.1001	$-2^{-3} + 2^{-5} - 2^{-7} + 2^{-9}$
h_{24}	0.3175	$2^{-2} + 2^{-4} + 2^{-8} + 2^{-10}$
h_{25}	0.5	2^{-1}

3.3 Scaling block

The maximum sine wave amplitude, which gives the maximum SNR, that can be given to the modulator is not the full scale of the quantizer. This amplitude termed as Maximum Stable Amplitude(MSA) is determined by the noise transfer function of the modulator. For the given modulator MSA was around 79.12% of the full scale. So signal swing at the output of modulator is (MSA)x(fullscale). Because of the high frequency noise the modulator output seems to be swing fully in four bits. Hence to get the correct full scale SNR, the signal should be restored to its full scale amplitude by scaling with a fixed number = 1/MSA after filtering the high frequency noise.

The scaling block is inserted at last in filter chain, to attenuate all the high frequency noise. The scaling number is greater than unity. Ideally the scaling value is 1/0.7912. But some margin should be accounted for passband ripples. If the passband gain is more than unity, retaining th same number of bits whine scaling can cause an overflow and hence distortion. This margin is determined by trial and error by giving a tone at various frequencies and ensuring that there is no overflow for this value at all frequencies. The scaling block also implemented using polyphase structure.

$$The \quad scaling \quad number = \frac{1}{0.8} = 1 + 2^{-2} = 1.25 \tag{3.11}$$

CHAPTER 4

Results and Conclusion

4.1 Simulation and results

The design and implementation details of the decimation filter in three stages using Hogenauer structure and polyphase decomposition are discussed in chapters 2 and 3. The actual circuit is modelled using the Hardware Description Language, Verilog and functional verification is done in Modelsim by writing appropriate test benches. Once this ideal circuit gives the same SNR obtained in Matlab for the same input stream, we can go ahead to the final steps of synthesizing and routing the circuit.

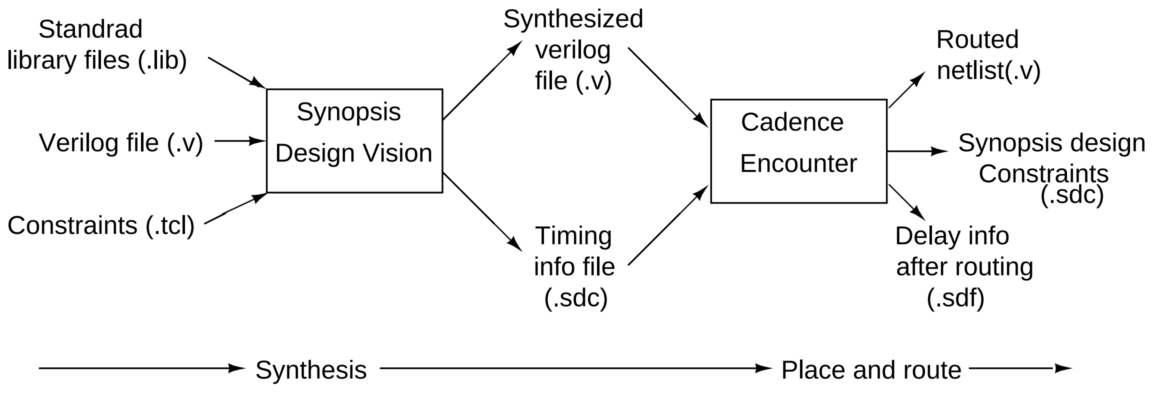


Figure 4.1: Block diagram of the design flow

The digital circuitry behaviorally modelled in verilog is to be mapped into actual circuit components based on standard cell libraries and constraints. This procedure is called synthesis and is performed using the Synopsis synthesizing tool *Design Vision*. Design vision gives the synthesized verilog file as output by reading three inputs (Figure 4.1).

1. Verilog code for the circuit

Table 4.1: Power report of the decimation filter

Module	**Input Clock**	**Power**	**% Power**
$SINC^4$ filter	1.536 MHz	20 μW	38
Halfband one	192 KHz	5.26 μW	10
Halfband two	98 KHz	15.8 μ W	30
Scaling Block	48 KHz	0.823 μW	1.5
Misc (clock dividers clock buffers)		10.593μW	20
Total		52.5 μW	100

2. Standard library files
3. Constraints

The verilog code is a behavioral description of the circuit. The standard cell libraries from UMC provide standard building blocks of the circuit in CMOS 180 nm technology. The constraints are given to the design vision using a .tcl script file.

- Minimize the area of the circuit
- Minimize the power consumption of the circuit
- Typical operating conditions with 1.8 V power supply.

Once the three input files (verilog, libraries and constraints) are ready, design vision compiles the input and creates a synthesized verilog file which is now based on standard building blocks. Design vision also gives a timing information file (.sdc). These files can be read using *Cadence encounter* to place and route (Figure 4.1). After placement and routing, the routed netlist can be saved as a verilog file and the timing information after routing can be extracted into a .sdf file both of which can be used for the post-route simulations.

The Value Change Dump (VCD) file is generated by the Modelsim for a given input vector. This switching activity file can be used for power analysis for that perticular input vector. *Synopsis Prime power* tool used for power calculation. Here input signal to $\Delta\Sigma$ modulator = 16 * 0.7912 * sin(2*π*f_{in}/f_s*n). No.of

Table 4.2: Variations of decimated signal amplitude and SNR with input tone frequency

Input signal frequency (KHz)	**Output signal amplitude** (Normalised)	**Output signal SNR** (in dB)
2.623	0.98	108.6
10	0.98	108
15	0.97	107.8
19	0.95	107.53

Table 4.3: Decimation filter design summary

Technology	0.18μm CMOS
Supply Voltage	1.8 V
Total power	52.5 μW
Area	236670μm^2 ($\approx$ 490 $\times$ 483)
Passband droop	-0.7519 dB at 21.6 KHz
Simulated SNR	108.6 dB (input=2.625$\acute{K}$Hz single tone)

simulation samples =2^{16} or duration of 42.6 ms. The power dissipation in each block is summarized in Table 4.1. At nyquist rate the output SNR will depends on amplitude of the decimated signal unlike in oversampled case where the SNR will depends on both time and amplitude. Scaling block is placed at last in decimator filter chain, opreates at nyquist rate. The variations of output signal amplitude and SNR with input signal frequency are summarised in Table 4.2.

The final simulation results are summarized in 4.3. From Table 4.1, it can be seen that most of the power is dissipated in SINC section and halfband two filter.

4.2 Conclusion

The total power consumed by the entire decimation filter is 52.5 μW. Thus hogenauer structure for sinc implementation and polyphase implementation for halfband filteres are proved to be very useful for low power decimation filters. In hogenauer structure pipelining and retiming will save 40% of total power. The low power of the halfband filter is attributed mainly to the decrease in the operating frequency of the circuit in polyphase. The decimation filter is completly designed with automated CAD tools, no handcraft circuits used.

CHAPTER 5

Feedback DAC error correction

A fully digital algorithm is described for acquiring and correcting the errors of the feedback DAC used in a multibit $\Delta\Sigma$ ADC. This method is highly accurate particularly for high speed modulators. In this method feedback DAC errors are determined by measuring the idle channel output and removing the DAC elements one by one[6]. This method requires simple computation, does not add excess loop delay, and requires no reconfiguration of the $\Delta\Sigma$ modulator. The errors so determined can be used to correct the output codes in the digital domain or the DAC elements in the analog domain. In this chapter, the basic principle and design of digital correction algorithm is discussed. The implementation details of the algorithm are explained in Chapter 6, and digitally correcting ADC output with Decimation filter and simulation results are demonstrated in Chapter 7.

5.1 Motivation

Multibit DS A/D converters are capable of a significantly higher in-band signal to quantization noise ratio (SQNR) compared to their single bit counterparts because they have an inherently smaller quantization step and they permit higher out of band gains[7]. They are also less prone to idle channel tones[7]. Additionally, in oversampling DS converters single bit DS results in a large amount of out-of-band quantization noise. Which requires relatively high-order analog filtering circuitry to attenuate the quantization noise. The use of a multi-bit D/A converter can significantly reduce this large amount of quantization noise, but care must be taken to ensure that the multi-bit converter remains linear.

5.1.1 Effect of DAC nonlinearity

The single biggest drawback of multibit modulators is the nonlinearity of the feedback DAC. Nonlinearity means the step size of the DAC will not be same for all input codes. The basic reason for this is, which ever way we realize the DAC whether, with bunch of parallel conductors or with current sources, they have device mismatch, causes non uniform step size. Figure 5.1 explains a two bit nonlinear DAC characteristics. Because of mismatches in DAC elements, output levels are different for each step size.

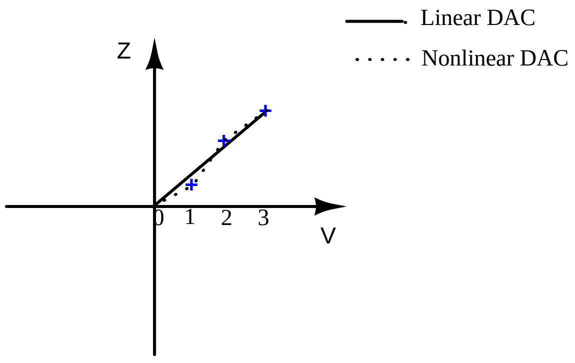


Figure 5.1: Example two bit nonlinear DAC. Solid curve shows ideal behaviour of the DAC while dotted curve shows nonlinear relation between input codes and output.

A DSM example with nonlinear DAC is shown in Figure 5.2(a). This is equivalent to injecting an error at loopfilter input. The nonlinearity of the DAC is modeled as a cascade of linear DAC and a nonlinear polynomial system, shown in Figure 5.2(b). The effect of DAC nonlinearity for a sinusoidal tone is shown in Figure 5.3, explains the increase in inband noise floor and harmonic distortion.

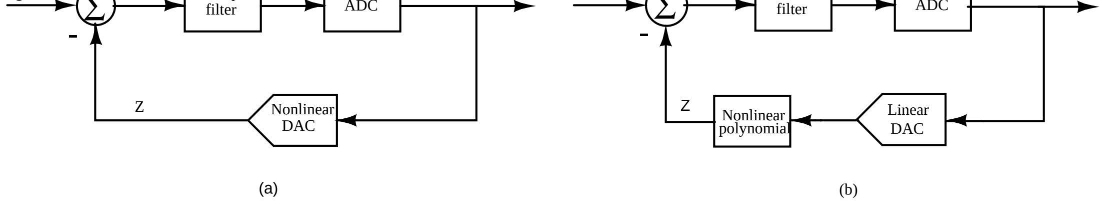


Figure 5.2: (a)Example DSM with nonlinear DAC. (b) Equivalent DSM with linear DAC followed by nonlinear polynomial system.

Dynamic element matching (DEM) is an effective and popular technique to convert the distortion into shaped noise and significantly improve the in band performance. However, this requires multiple stages of gates or switches in the digital feedback path. For example, data weighted averaging (DWA) for a 4 bit DAC requires a four stage barrel shifter driven from the accumulated input. Higher order dynamic element matching schemes require more complex computations. For high speed modulators the delay introduced by the DEM logic can be a significant fraction of the sampling period and is best avoided.

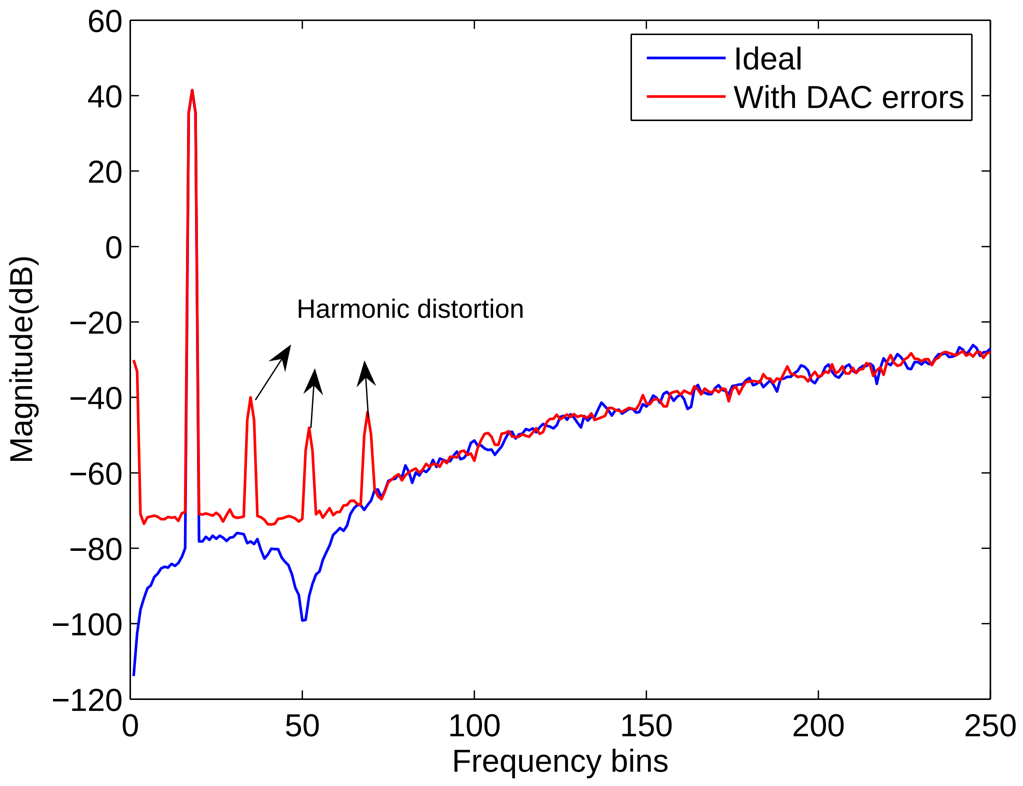


Figure 5.3: Comparison of nonlinear DSM Output spectrum with linear DSM($OSR = 32, OBG = 3, No.of bits = 4$).

An alternative is digital correction of feedback DAC errors as shown in Figure 5.4. Errors in DAC output values from the ideal values are measured, stored digitally, and added to the output of the modulator so that the final digital output accurately corresponds to the analog output of the DAC. In this case, the correction circuitry is outside the loop and does not contribute to excess delay.

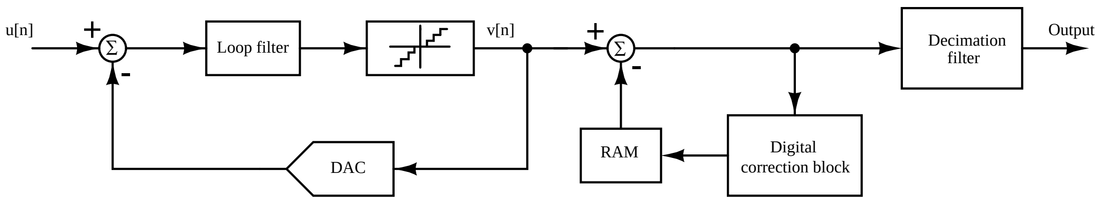


Figure 5.4: Multibit ΔΣ modulator with digital correction

5.2 Digital correction algorithm

The digital correction algorithm is implemented for the ΔΣ modulator explained in section 2.3. This algorithm is independent to the type of the DAC elements-current sources, resistors, or capacitors. A prototype resistive DAC system is assumed for the ΔΣ modulator shown in Figure 5.5. The following assumptions are made for this system.

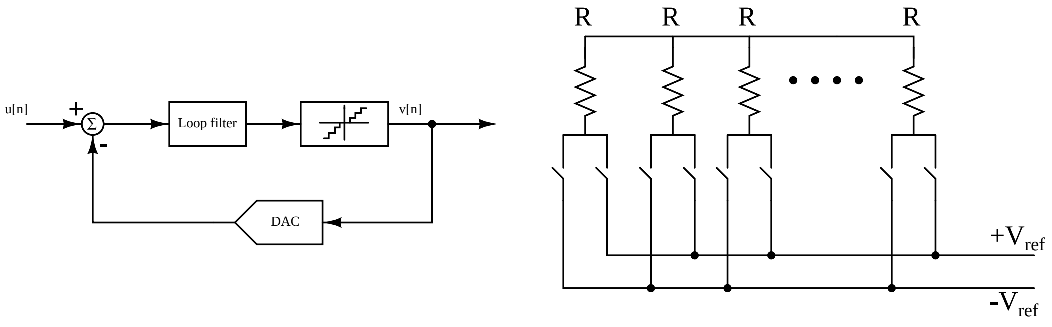


Figure 5.5: Multibit ΔΣ modulator with a prototype resistive DAC

- The resistive DAC elements are switched in a complementary fashion. That means, based on the digital code, the voltage source V_{ref} in a resistive DAC provides a feedback current of $+V_{ref}/R$ or $-V_{ref}/R$, shown in Figure 5.5. This is the case with most implementations.
- Each dimensionless DAC elements with a nominal unit value of 1 and actual values (with DAC nonlinearity) of $D_{0...N-1}$ are considered.

With N elements, the DAC output has N+1 possible values with -$\sum_{k=0}^{N-1} D_k$ (nominally -N) for the smallest input code and $\sum_{k=0}^{N-1} D_k$ (nominally +N) for the largest input code. With each increment in the input code, one of the units switches from negative to positive with LSB of 2.

For the ΔΣ modulator having four bit output will have 15 DAC elements. I.e $D_{0,1...14}$. Ideally minimum value is -15 for smallest input code and +15 for the largest input code. With DAC nonlinearity the output value will slightly differ from the accurate. Assume that the erroronious DAC outputs for each input code are $E'_0, E'_1, ..., E'_{15}$.

$$\begin{aligned}
E'_0 &= -D_0 - D_1 - D_2 - D_3 - \cdots\cdots - D_{14} \\
E'_1 &= +D_0 - D_1 - D_2 - D_3 - \cdots\cdots - D_{14} \\
E'_2 &= +D_0 + D_1 - D_2 - D_3 - \cdots\cdots - D_{14} \\
E'_3 &= +D_0 + D_1 + D_2 - D_3 - \cdots\cdots - D_{14} \\
&\quad . \\
&\quad . \\
&\quad . \\
&\quad . \\
E'_{15} &= +D_0 + D_1 + D_2 + D_3 + \cdots\cdots + D_{14}
\end{aligned} \tag{5.1}$$

Ideally all the DAC elements should be equal. For example, E'_0 equal to $-15D_0$. One way to make all DAC elements ideal is to write DAC outputs for each input code in terms of only one DAC element. Subtracting D_k-D_0 from D_k for $k = 1 \ldots N-1$ make all the DAC units equal to D_0. Now correction values for each input code becomes

$$\begin{aligned}
E_0 &= 0 - (D_1 - D_0) - (D_2 - D_0) - (D_3 - D_1) - \cdots\cdots - (D_{14} - D_0) \\
E_1 &= 0 - (D_1 - D_0) - (D_2 - D_0) - (D_3 - D_1) - \cdots\cdots - (D_{14} - D_0) \\
E_2 &= 0 + (D_1 - D_0) - (D_2 - D_0) - (D_3 - D_1) - \cdots\cdots - (D_{14} - D_0) \\
E_3 &= 0 + (D_1 - D_0) + (D_2 - D_0) - (D_3 - D_1) - \cdots\cdots - (D_{14} - D_0) \\
&\quad . \\
&\quad . \\
&\quad . \\
E_{15} &= 0 + (D_1 - D_0) + (D_2 - D_0) + (D_3 - D_1) + \cdots\cdots + (D_{14} - D_0)
\end{aligned} \tag{5.2}$$

Now subtracting E_k from E'_k for $k = 0, 1, ..15$ will give linearised outputs for each input code. $B_{0,1,2....,15}$ are the DAC outputs for each input code

$$\begin{aligned}
B_0 &= E'_0 - E_0 = -15D_0 \\
B_1 &= E'_1 - E_1 = -13D_0 \\
B_2 &= E'_2 - E_2 = -11D_0 \\
B_3 &= E'_3 - E_3 = -9D_0 \\
&. \\
&. \\
&. \\
B_{14} &= E'_{14} - E_{14} = +13D_0 \\
B_{15} &= E'_{15} - E_{15} = +15D_0 \qquad (5.3)
\end{aligned}$$

In general, for N elements, the DAC output has N+1 possible output values, so the required correction matrix is given in below.

$$\begin{bmatrix} E_0 \\ E_1 \\ E_2 \\ E_3 \\ . \\ . \\ . \\ E_N \end{bmatrix}_{(N+1)\times 1} = \begin{bmatrix} -1 & -1 & -1 & -1 & \cdots -1 \\ -1 & -1 & -1 & -1 & \cdots -1 \\ +1 & -1 & -1 & -1 & \cdots -1 \\ +1 & +1 & -1 & -1 & \cdots -1 \\ & & & . & \\ & & & . & \\ & & & . & \\ +1 & +1 & +1 & +1 & \cdots +1 \end{bmatrix}_{(N+1)\times N} \begin{bmatrix} 0 \\ (D_1 - D_0) \\ (D_2 - D_0) \\ (D_3 - D_0) \\ . \\ . \\ . \\ (D_{N-1} - D_0) \end{bmatrix}_{N\times 1}$$

To implement this technique, the relative errors between DAC elements must be available in digital form. A possibility is to reconfigure the loop as a single bit modulator[8] and measure all the elements of the multibit DAC against a single element. But this reconfiguration requires modifying the loop filter because a single bit modulator is not stable with the high OBGs typical of multibit modulators, resulting in an overhead in the design of the loop filter and additional simulation effort for verifying the circuit in all modes.

To overcome these shortcomings, a method for determining DAC errors using minimal modifications to the DS modulator is investigated. It is based on observing the output of the modulator with the input set to zero and DAC elements removed one at a time.

5.3 Determination of feedback DAC errors

The mismatch between DAC current sources is estimated from idle channel measurements. In the idle state, the quantizer output switches between a few levels, typically three to five, in the middle of the range. For $\Delta\Sigma$ modulator given, having 15 DAC units $D_{0...14}$, elements $D_{0,1..5}$, $D_{9,10...14}$ are not switched in the idle state.

To explain this, a example $\Delta\Sigma$ modulator system is taken with same specifications listed in Table 2.1. The non-linearity of the DAC elements is included in Schrier's $\Delta\Sigma$ tool box for simulations. The output of the nonliner DSM for zero input is shown in Figure 5.6. The output of the DSM is changing between -3 to 3 for zero input(*this refers to -15 to 15 scale where, LSB=2*).

To know the status of DAC elements switching, a *transform matrix* which, gives DAC output values for each input code is used. Refer 4×4 matrix from Table 5.1. For the change in input codes from -3 to 3, all the DAC elements are unchanged expect $D_{6,7,8}$. Hence for zero input to the DSM, the switching happens in $D_{6,7,8}$ levels.

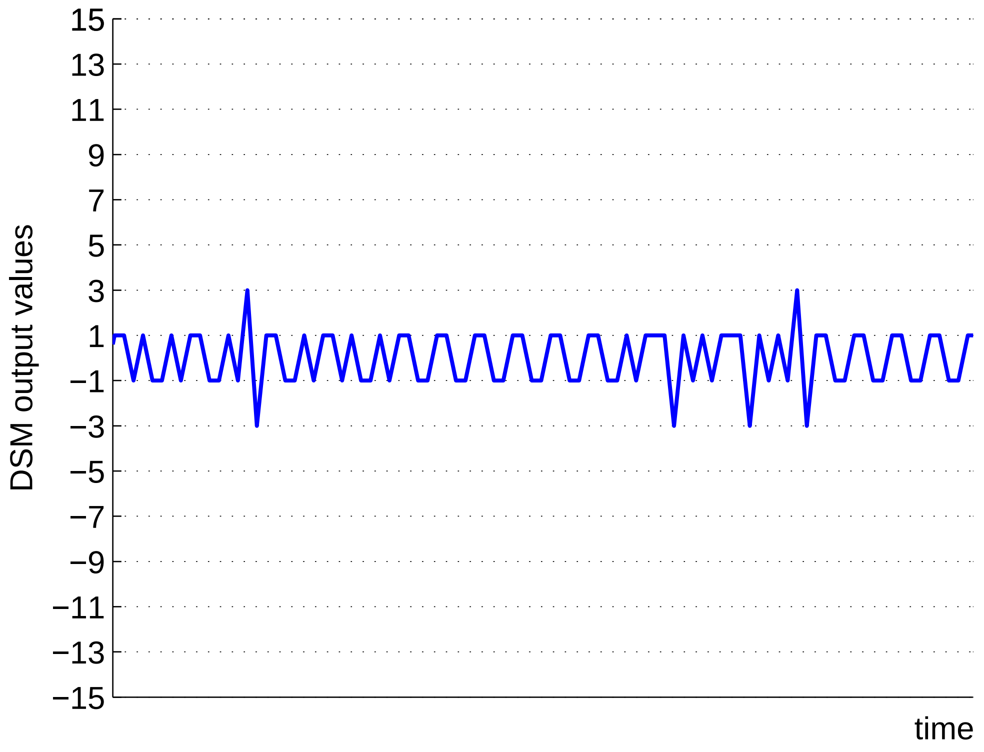


Figure 5.6: Output of DSM(having nonlinear DAC elements) for zero input. Output is switching between -3 to 3

Table 5.1: Transform matrix, which gives DAC output values for each input code.

	DAC elements	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14
Input codes																
-15		-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-13		+1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-11		+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-9		+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-7		+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-5		+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-3		+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-1
-1		+1	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1
1		+1	+1	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1
3		+1	+1	+1	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1
5		+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1
7		+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1
9		+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-1	-1	-1
11		+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-1	-1
13		+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-1
15		+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1

The technique presented here is based on observing the DSM digital output with DAC units removed one at a time. When a particular DAC element is removed, its contribution to the DAC output is zero, instead of being ±1. The digital output V is simply the number of DAC elements that have a positive weight. With zero inputs, the modulator forces the *average* output of the DAC to be D_{off} , an offset in terms of DAC units which is used to represent the effect of the offset due to the offset of the loop filter and imperfect disabling of the input. The digital output of the modulator, which is the input to the DAC represents the sum of D_{off} and the average nonlinearity contributed by the DAC elements at that particular output value.

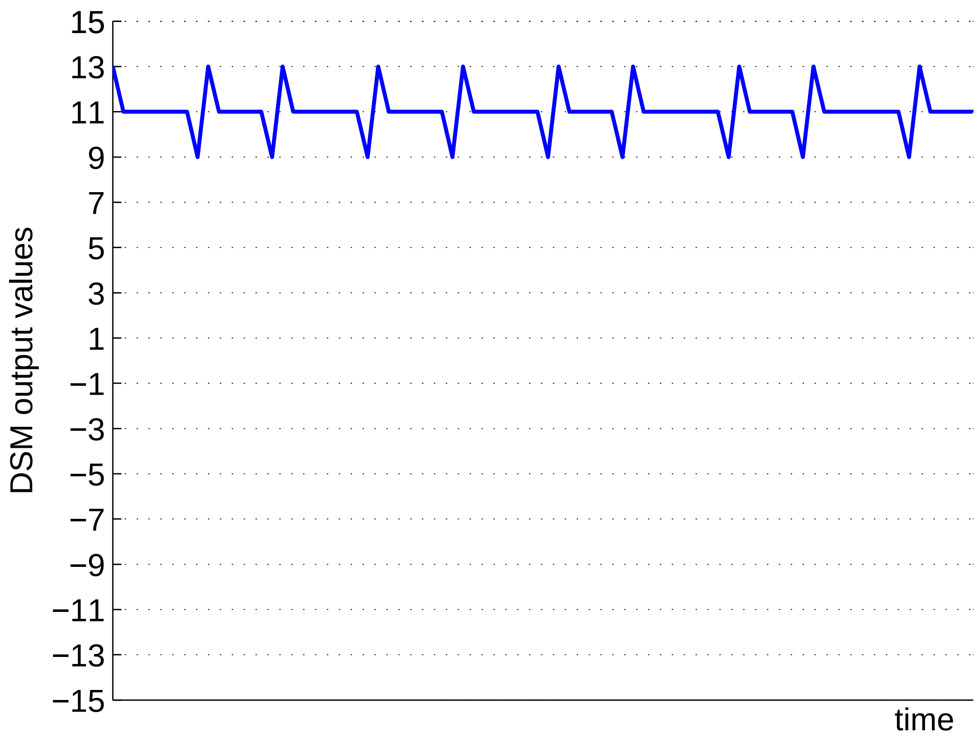


Figure 5.7: ΔΣ modulator output for input dc offset of 11.

A dc offset of $5.5\times LSB$=11 is added to the input to shift the switching activity to different elements of the DAC. For dc offset of 11, the output of the DSM will varying from 9 to 13 as shown in Figure 5.7. From Table 5.1 for DAC inputs 9,11 and 13, DAC elements $D_{0,1..11}$, D_{14} are in idle state. $D_{0,1..11}$ contribute with a positive sign and D_{14} contribute with a negative sign.

First, remove D_0 from the circuit, shown in Figure 5.8(a). When an element is removed, its contribution is zero. Since D_0 does not change its sign, and the DAC output is subtracted from the input, this is equivalent to providing an input of D_0 to the modulator, Figure 5.8(b). The average digital output $V_{avg,0}$ equals D_0+ D_{off} $+NL|_{D_0+D_{off}}$, where $NL|_{D_0+D_{off}}$ is the input referred nonlinearity of the DAC at an average output of D_0+D_{off}.

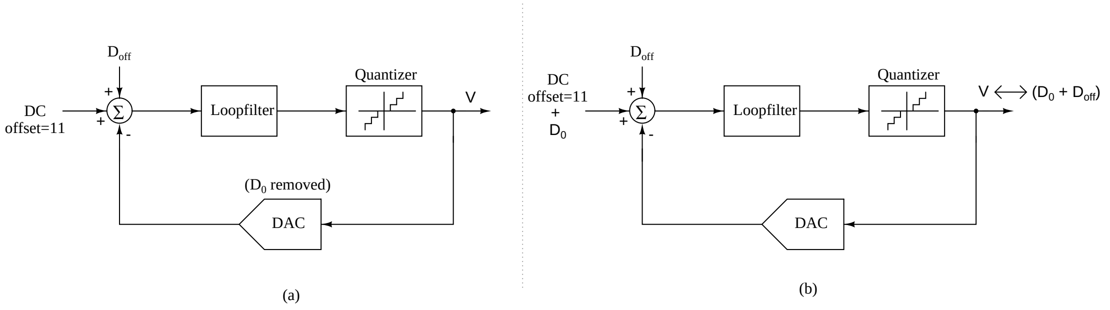


Figure 5.8: (a)$\Delta\Sigma$ modulator with D_0 element removing from the DAC, (b) Equivalent picture with an input of D_0+DC offset to the modulator and the DAC elements remain unchanged

Next, remove D_1 from the circuit. This is equivalent to providing an input of D_1 to the modulator. The average digital output V_{avg}, 1 equals D_1+D_{off}+$NL|_{D_1+D_{off}}$, where $NL|_{D_1+D_{off}}$ is the input referred nonlinearity of the DAC at an average output of D_1+D_{off}. Since D_0 and D_1 differ by a very small amount, the nonlinear contributions $NL|_{D_1+D_{off}}$ and $NL|_{D_0+D_{off}}$ can be considered identical to a first order. Thus, the difference between D_0 and D_1 can be calculated very simply by taking the difference between the corresponding average digital outputs. Similarly difference between successive units from D_1 to D_{11} can be calculated.

$$D_{k+1} - D_k = V_{avg,k+1} - V_{avg,k} \quad k = 0...10 \tag{5.4}$$

The differences D_{12}-D_{11} to D_{14}-D_{13} still need to be calculated. For this, a dc offset of 2.5 $\times$ LSB=5 is added to the input to shift the switching activity to different elements of the DAC. The output of the DSM is changing between 3 to 7 for dc offset 5, shown in Figure 5.9. From Table, 5.1 the switching DAC elements are D_9 and D_{10}. So for the 16 level DAC , adding an offset of 5 (2.5 LSB offset) causes the sources D_9 to D_{10} to be switched instead of D_{12} to D_{13}. This enables calculation of differences D_k-D_{k-1} for $12 \leq k \leq 14$ using the method outlined above.

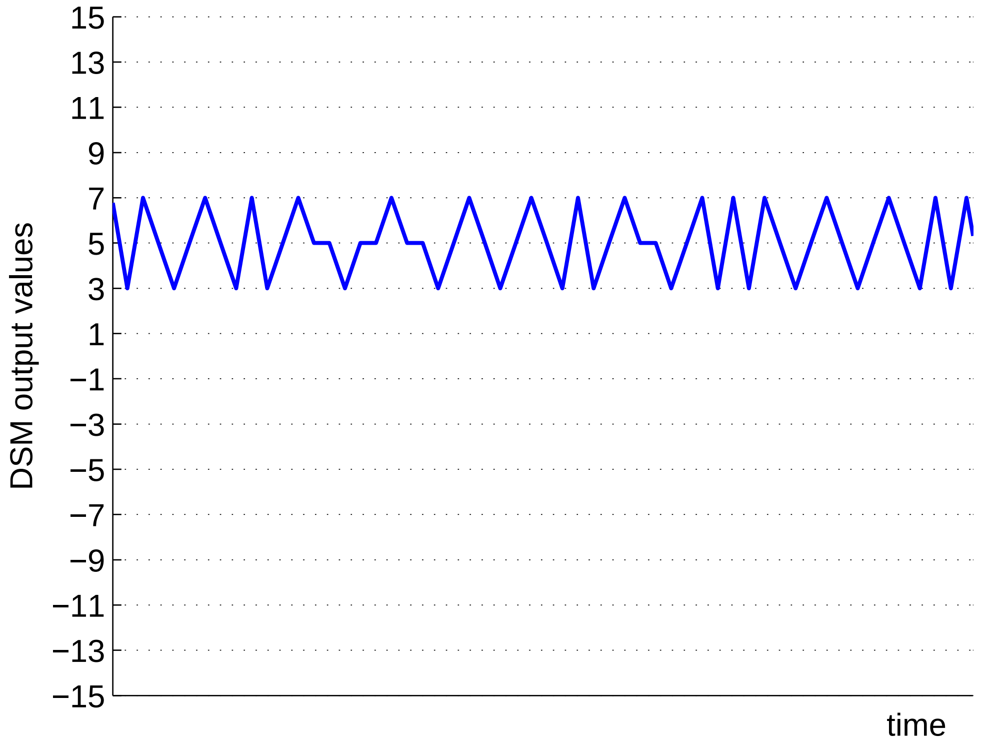


Figure 5.9: $\Delta\Sigma$ modulator output for input dc offset of 5.

The method outlined earlier result in the difference in the values of successive DAC elements. The errors of all DAC elements are computed with respect to one of the elements, say D_0, by appropriately summing the differences between successive elements.

$$D_k - D_0 = (D_k - D_{k-1}) + (D_{k-1} - D_{k-2}) + ... + (D_1 - D_0) \tag{5.5}$$

As explained in previous section these errors of all DAC elements are computed with respect to one of the elements D_k-D_0 for k=0,1..14, are used in matrix given below to get correction values $E_0, E_1, ...E_{15}$ (for output codes 0,1. . .15

respectively).

$$\begin{bmatrix} E_0 \\ E_1 \\ E_2 \\ \cdot \\ \cdot \\ \cdot \\ E_{15} \end{bmatrix}_{16\times 1} = \begin{bmatrix} -1 & -1 & -1 & \cdots - 1 \\ -1 & -1 & -1 & \cdots - 1 \\ +1 & -1 & -1 & \cdots - 1 \\ & & & \cdot \\ & & & \cdot \\ & & & \cdot \\ +1 & +1 & +1 & \cdots + 1 \end{bmatrix}_{16\times 15} \begin{bmatrix} 0 \\ (D_1 - D_0) \\ (D_2 - D_0) \\ \cdot \\ \cdot \\ \cdot \\ (D_{14} - D_0) \end{bmatrix}_{15\times 1}$$

CHAPTER 6

Implementation

The top level architecture for implementing digital correction algorithm is shown in Figure 6.1. The three steps involved in digital correction algorithm process are, obtaining the relative errors between DAC elements, finding the errors of all DAC elements with respect to one of the elements and obtaining the correction values. These steps are done one after other, hence same hardware can be used for all steps.

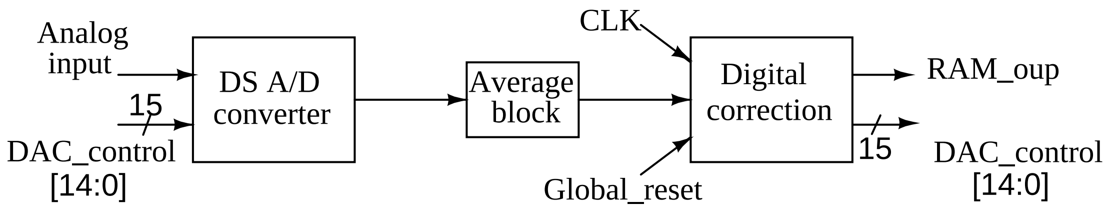


Figure 6.1: Top level architecture for implementing digital correction algorithm.

Digital correction operation starts with posedge of Global_reset signal which resets all registers in the block. The detail architecture of digital correction block shown in Figure 6.2. The actual architecture is modeled using Hardware Description language, verilog. To follow each step in algorithm, different serial control signals are generated. Each signal performs the different operations in time. These control signals are step,step0,step1,step2,step3,step4 and step5.

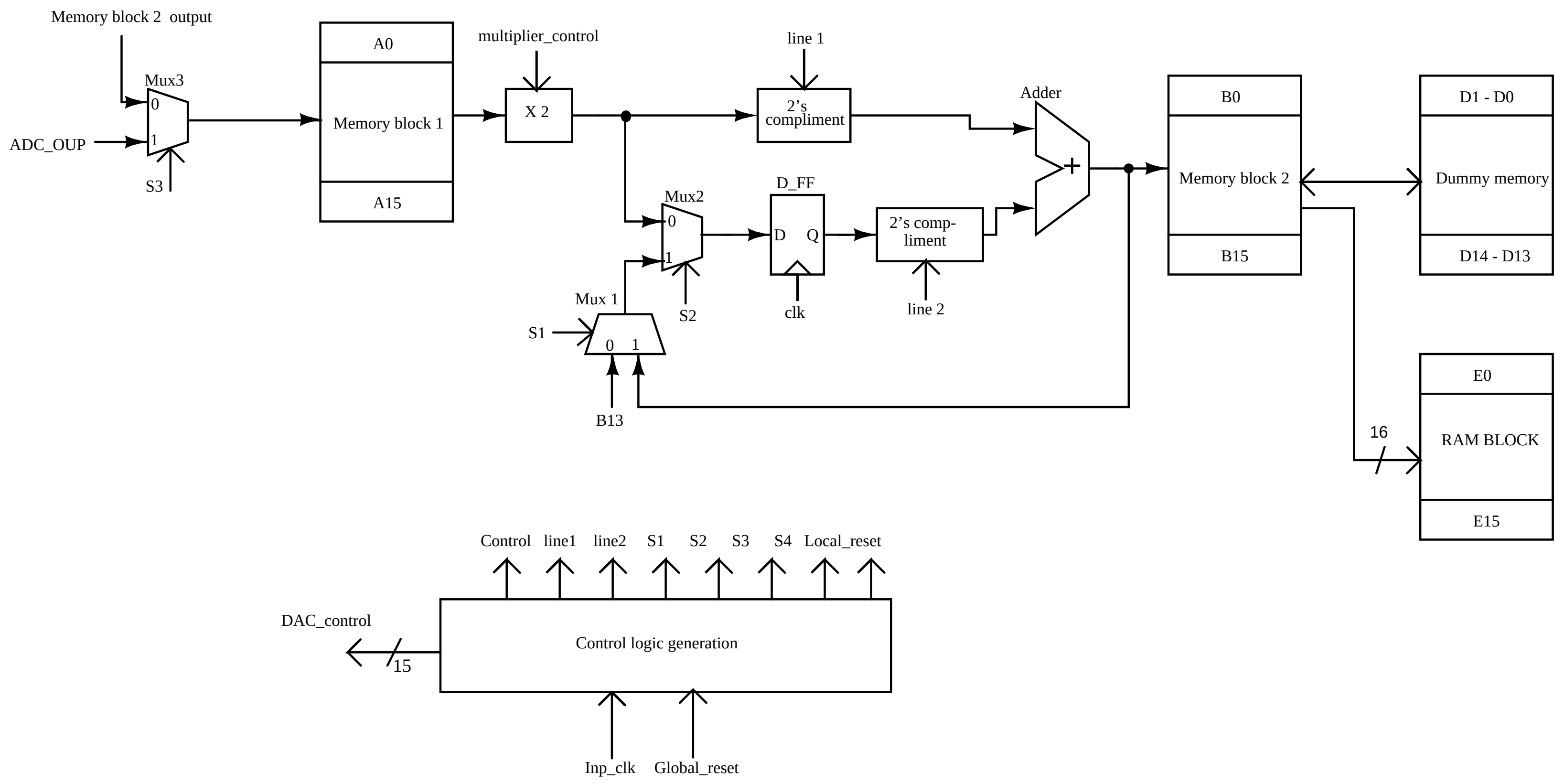


Figure 6.2: Detail architecture of digital correction mechanism.

Phase1 : Obtaining DAC element errors $D_{0,1,...,14}$

The time period of the clock common with modulator, is 0.651 μS. Clock generator modules are used to slow down the clock. As Global_reset given to digital calibration block, the 15bit DAC_control signal reset to 111111111111111, means all DAC elements are unchanged. With the *step* signal active high, DAC_control goes to 111111111111110 which remove the DAC element D_0. As explained in previous chapter with D_0 removed, the *average* digital output of DSM is $V_{avg,0}$. To average the output of the DSM, a average block with appropriate bit size is used is shown in Figure 6.1. The number of samples (N) for averaging depends on the desired accuracy after correction and total quantization noise of the modulator. Here N=2^{16}. This averaged value corresponds to the error of DAC element D_0, is stored in A0 location of Memory block1.

After N number of cycles the DAC_control changes to 111111111111101, means second DAC element D_1 is removed while remaining DAC elements are unchanged. With this setup, the average value $V_{avg,1}$, error corresponds to DAC element D_1 is stored in A1 location of Memory block1.

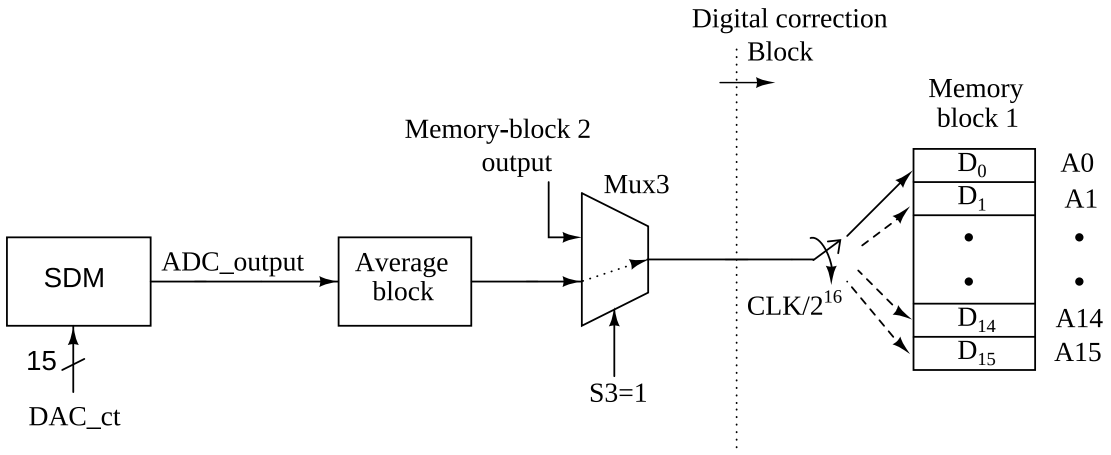


Figure 6.3: Determing each DAC element error by removing one DAC element at a time. Setup shows status of the different control signals.

This process is continued to get all DAC element errors $D_{2,3,..,14}$ and stored in corresponding memory locations A2,A3,...,A14. The total setup explained in Figure 6.3. This process will take 15×2^{16} cycles to complete. Different control signals used in this architecture are

Table 6.1: Status of each control signal during different phases.

	line1	**line2**	**Mux1 control(S1)**	**Mux2 control(S2)**	**Mux3 control(S3)**	**multiplier-control**
phase1	-	-	-	high	-	
phase2	low	high	high	low	low	low
phase3	low	low	high	high	low	low
phase4(a)	high	low	high	high	low	low
phase4(b)	low	low	low	high	low	high

- Multiplier_control, when low it disables the multiplication by 2 operation.
- line1, line2 are control signals which when high, activates the two's compliment block.

The status of each control signal during different phases is tabulated in Table 6.1.

Phase2 : Determining mismatch between successive DAC elements

This process starts with active high *step0* signal. During this phase, line1, Mux3-control S3 and multiplier_control are active low while line2 is active high. Select line S2 is kept high so that, D_FF will take Memory-block1 output. The DAC element errors present in Memory-block1 are transfered one by one to Memory-block2 such that, the successive DAC elements differences, $D_{k+1} - D_K$ for k=0,1,..13 are stored in B0,B1,...B13 memory locations respectively. This process is explained in Figure 6.4.

These stored error values are calculated for a dc offset of 11. Before redoing these two steps for a dc offset of $2.5 \times LSB$, the successive DAC elements differences, $D_{k+1} - D_K$ of dc offset 11 are transfered from Memory-block2 to a Dummy-memory.

Phase3 : Errors of all DAC elements with respect to D_0 element

The process of calculating errors with respect to one element is shown in Figure 6.5. With the active high of control signal *step1*, the successive differences of

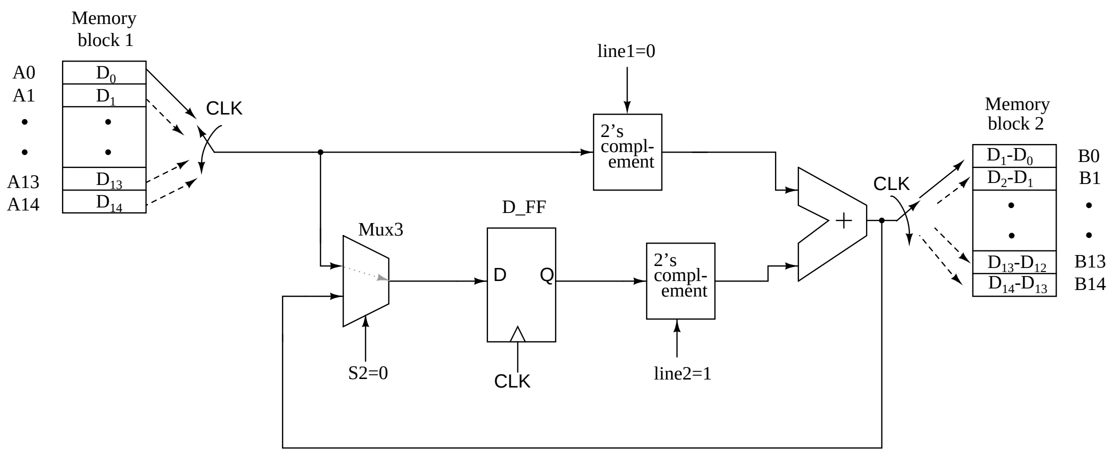


Figure 6.4: Determining mismatch between successive DAC elements. Setup shows status of the different control signals for this phase.

DAC elements $D_{k+1} - D_K$ ($0 \leq k \leq 10$) from Dummy-memory and D_{12}-D_{11} to $D_{14}D_{13}$ are from B11,B12,B13,B14 locations of Memory-block2 are transfered to A0,A1,...A13 locations of Memory-block1.

During same phase, the errors of all DAC elements with respect to D_0 element, $D_{k+1} - D_0$ (for k=0 to 13) are calculated. During the second clock cycle of this phase $D_2 - D_1$ is transfered from second location of Dummy-memory to A1 location in Memory-block1, at the same time $D_1 - D_0$ from A0 location of Memory-block1 is transfered to B0 location of Memory-block2. In the third clock cycle $D_2 - D_1$ from A1 location of Memory-block1 is added with previously transfered $D_1 - D_0$ to get $D_2 - D_0$ and it is stored in B1 location of Memory-block2. The value $D_2 - D_0$ is added with $D_3 - D_2$ successively to get $D_3 - D_0$. So errors of all DAC elements with respect to D_0 element, $D_{k+1} - D_0$ (for k=0 to 13) are calculated and stored in Memory-block2.

Phase4(a) : Correction value E_0

This phase activates by the active high of the control signal *step2*. From the equation 5.2 the correction value E_0 is

$$E_0 = 0 - (D_1 - D_0) - (D_2 - D_0) - (D_3 - D_1) - \cdots\cdots - (D_{14} - D_0) \qquad (6.1)$$

In this phase the errors of all DAC elements with respect to D_0 element,

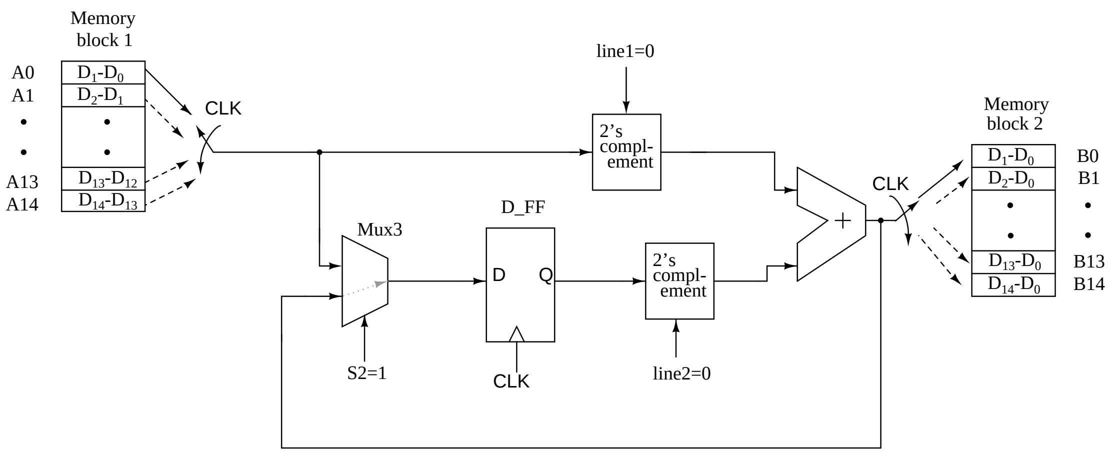


Figure 6.5: Determining mismatch DAC elements with respect to D_0 element. Setup shows status of the different control signals for this phase.

$D_{k+1} - D_0$ (for k=0 to 13) are transfered from Memory-block2 to Memory-block1. During the second clock cycle of this phase the error $D_1 - D_0$ from A0 location of Memory-block1 is transfered to Memory-block2. In the third clock cycle, the error $D_2 - D_0$ used to obtain cumulative addition $-(D_1 - D_0) - (D_2 - D_0)$ and it is stored in B1. As this process goes on, the cumulative value of all $D_{k+1} - D_0$ (for k=0 to 13) errors which is E_0 is stored in B13.

Phase4(b) : Obtaining remaining correction values

Using E_0, and $D_{k+1} - D_0$ values stored in Memory-block1, the remaining correction values are given in equation 6.2. This phase of calculating correction values activates with *step3* control signal.

$$\begin{aligned}
E_1 &= E_0 \\
E_2 &= E_1 + 2 * (D_1 - D_0) \\
E_3 &= E_2 + 2 * (D_2 - D_0) \\
E_4 &= E_3 + 2 * (D_3 - D_0) \\
E_5 &= E_4 + 2 * (D_4 - D_0) \\
E_6 &= E_5 + 2 * (D_5 - D_0) \\
E_7 &= E_6 + 2 * (D_6 - D_0) \\
E_8 &= E_7 + 2 * (D_7 - D_0) \\
E_9 &= E_8 + 2 * (D_8 - D_0) \\
E_{10} &= E_9 + 2 * (D_9 - D_0) \\
E_{11} &= E_{10} + 2 * (D_{10} - D_0) \\
E_{12} &= E_{11} + 2 * (D_{11} - D_0) \\
E_{13} &= E_{12} + 2 * (D_{12} - D_0) \\
E_{14} &= E_{13} + 2 * (D_{13} - D_0) \\
E_{15} &= E_{14} + 2 * (D_{14} - D_0) \qquad (6.2)
\end{aligned}$$

The setup for calculating error correction values from $D_{k+1} - D_0$ for $K = 0, 1 \ldots 15$ is shown in Figure 6.6. The error correction values are transfered from Memory-block2 to a 16×16 register set, RAM block.

Four sets of registers are required for digital correction algorithm. Two sets are to store difference between successive values, difference with respect to a single element, one set is to store difference between successive values for initial dc offset and RAM to correction values for each output code. Once the errors are determined and the correction values stored, first three register sets are inactive. The correction values are added directly to the digital output of $\Delta\Sigma$ modulator as shown in Figure 6.1.

The number of cycles required for averaging depends on the desired accuracy

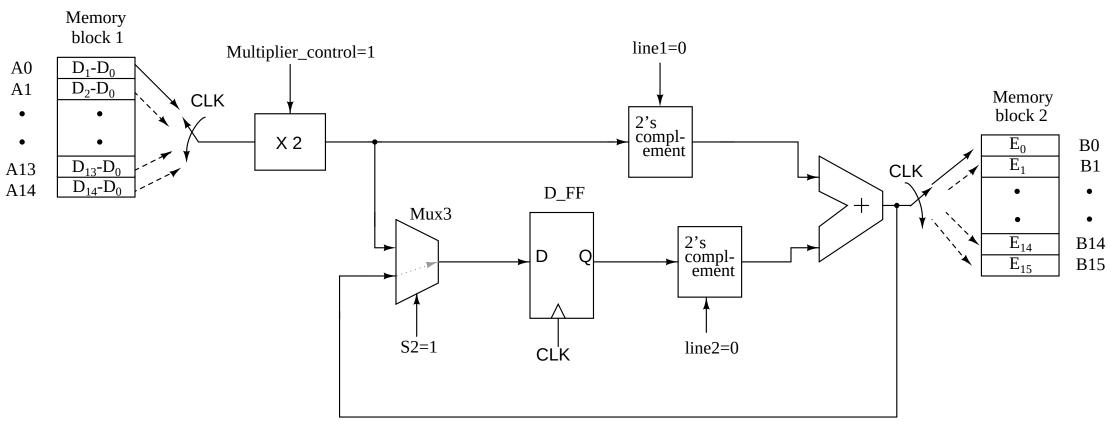


Figure 6.6: Calculating error values from $D_{k+1} - D_0$ for $K = 0, 1 \ldots 15$. Setup shows status of the different control signals for this phase.

after correction and total quantization noise of the modulator (which has to be filtered by averaging). Once a desired degree of filtering is achieved, the accuracy to which the DAC units are measured is limited by the nonlinearity of the DS modulator (i.e. $NL|_{D_0+D_{off}}$ and $NL|_{D_1+D_{off}}$ are not quite identical as assumed. The correction values obtained by the first correction can be used to linearise the modulator. The process of removing DAC elements one by one and measuring successive differences can be repeated with this improved modulator. In this manner, the correction values can be progressively refined. The correction loop is shown in Figure 5.4. For first correction loop all RAM contents are reset to zeros and after phase4(b) the correction values are stored in RAM, which are used for second correction loop.

CHAPTER 7

Results and Conclusion

The Digital correction algorithm is modelled using the Hardware Description Language, Verilog and functional verification is done in Modelsim by writing appropriate test benches. Design-vision used for synthesizing and Cadence Encounter for place & route the design.

The block diagram of digital correction algorithm with register sizes is shown in Figure 7.1. From the simulations, a 16 bit register is sufficient to store the correction values. As the number of samples for averaging(N) is 2^{16}, correction values should be normalised with 2^{16} to get the absolute error before adding to ADC output. During second correction, to improve the resolution of average of the correction values N value chosen as 2^{18}.

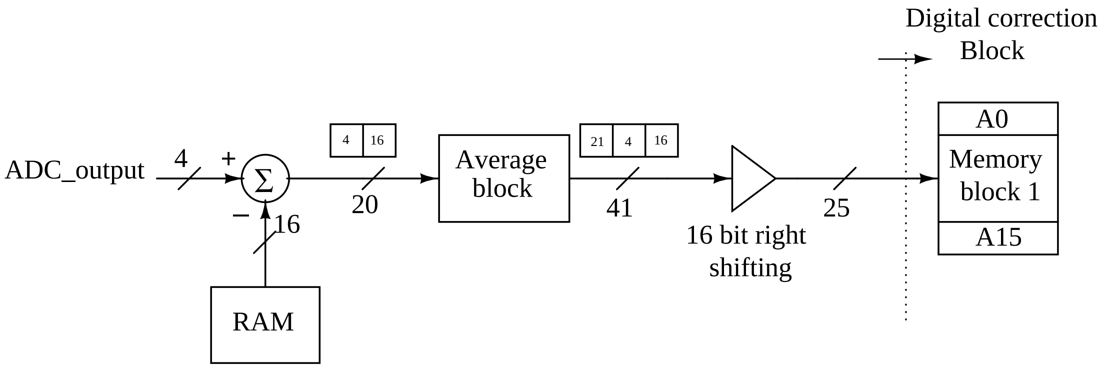


Figure 7.1: Block diagram for second correction

Even for 2^{18} average samples the RAM size is 16×16, that means before transferring from Memory-block2 the correction values are divided by 2^2. A two bit programmable pin is included in the design, facilitates to vary the number of averaged samples for second correction. For the first correction this value is fixed to 2^{16}. When programmable pin P=00, N for second correction is 2^{17} and N will become twice with one bit increase in P. So in Figure 7.1, the register width required for the average block for maximum average number of samples ($N = 2^{20}$)

is 4+16+21=41. After average block, to get required inband SNR, the sufficient number of bits are 25. So the output of average block is right shifted by 16 bits to get a 25 bit output. Hence the three memory block sets, memory-block 1, memory-block 2 and dummy-memeory required to have 25 bit size each.

After getting the correction values in RAM location, the correction circuitry is inactive. Except RAM, for all blocks clock is disconnected. The output spectrums of $\Delta\Sigma$ modulator with the digital correction algorithm are shown in Figure 7.2. After first correction the noise level is nearly down to the ideal level. Harmonic distortion also reduced significantly, but still some distortion components are presented. The output spectrum for correction values determined for second time with improved modulator, has noise and distortion close to ideal values. The results are summarized in Table 7.1.

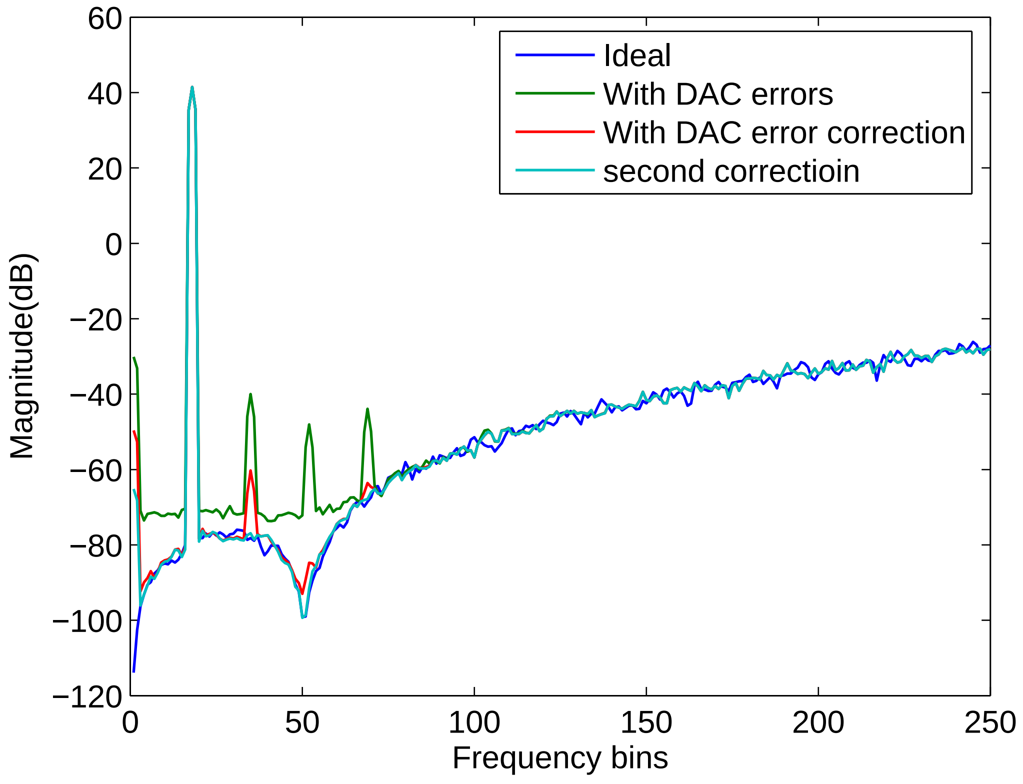


Figure 7.2: $\Delta\Sigma$ modulator output spectrum showing performance improvement with digital correction.

With the programmable switch the number samples for averaging(N) are varied. Figure 7.3 shows the output spectrum for different values of N. Clearly

Table 7.1: Results for DSM having $OSR = 32, OBG = 3$, 16 levels

	SNR	HD_2	HD_3
Ideal	103.01 dB	-	-
With DAC errors	95.26 dB	-81.4 dB	-89.5 dB
First correction	102.59 dB	-101.5 dB	-125.5 dB
Second correction	102.73 dB	-118.4 dB	-133.0 dB

as the number of samples for averaging increases, the harmonic distortion goes down. Synthesis results are summarized in Table 7.2. The power consumed during calibration phase is 185μW. Once the errors are calulated correction values are stored, except RAM all blocks are inactive. In correcting mode power consumed is 11.5μW in RAM block.

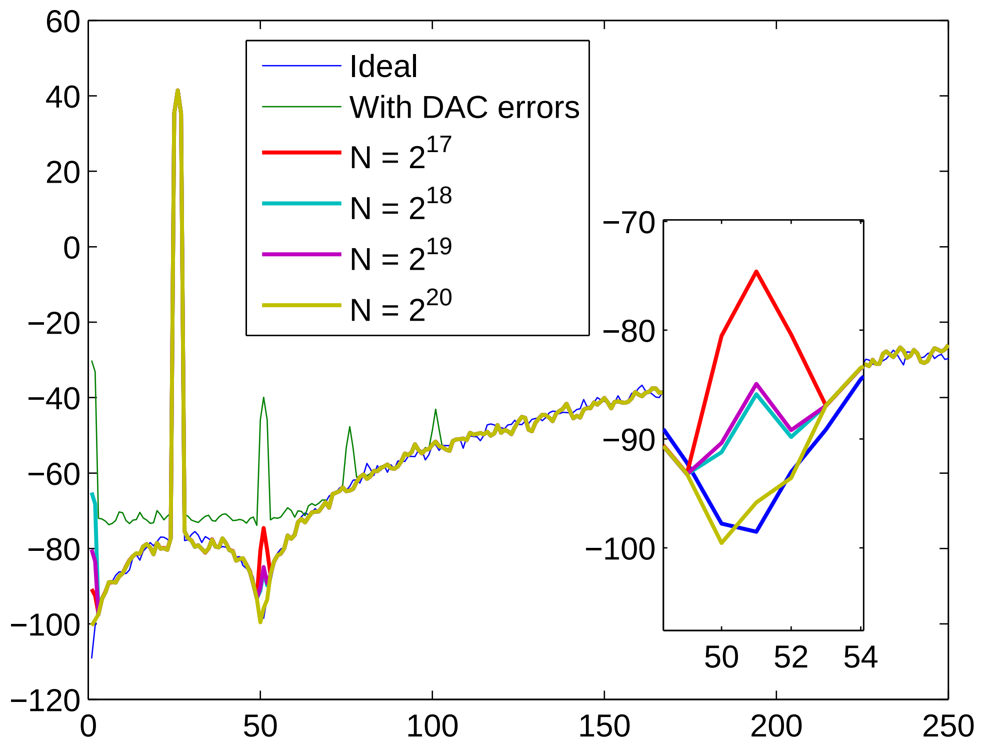


Figure 7.3: $\Delta\Sigma$ modulator output spectrum showing decrease in harmonic distortion with increase in N value.

Table 7.2: Summary of digital correction design

Technology	0.18μm CMOS
Supply Voltage	1.8 V
Total power	185μW
Core area	235654μm^2 ($\approx 493 \times 478$)

7.1 Digital correction with Decimation filter

The stored digital correction values are added to the output of the modulator so that the final digital output accurately corresponds to the analog output of the DAC. This digital correction process is combined with the decimation filter. That means, the corresponding correction value of each output code of DSM is added during decimation process.

As the correction values have 16 bit decimal positions, to get the absolute error, correction value should be normalised with 2^{16} before adding to input codes. For this we need to right shift the input data by 17 bits. Figure 7.4 shows one way to add the correction values to the original data. After CIC section, input data is multiplied by 2^9 and correction values are divided by 2^8 before subtracting the error values. In this structure CIC filter sections, CIC1 and CIC2 require 16 bit and 28 bit registers to avoid overflow, which increases both hardware and power consumption.

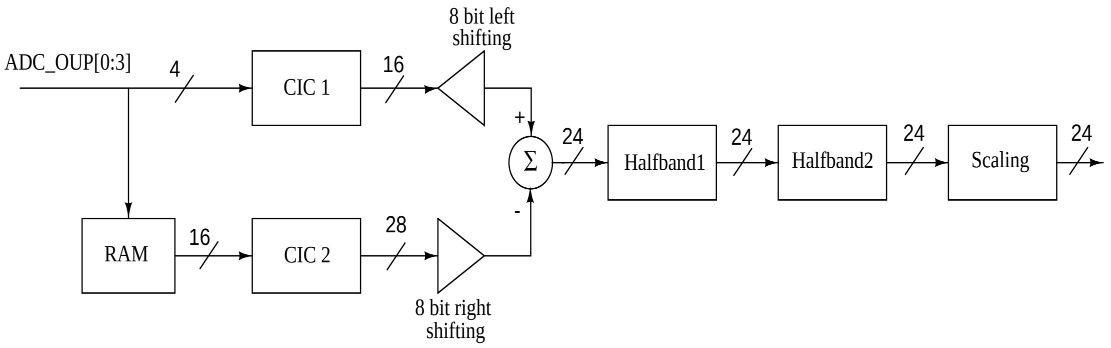


Figure 7.4: Digital correction scheme with decimator, showing correcting data at output of CIC filter section.

Other scheme where correction values are added before CIC section is shown in Figure 7.5. The input is multiplied by 2^{14} and correction values are divided by 2^3, makes a 18 bit corrected input given to decimation filter. This structure requires a 30 bit register for CIC section which means approximately (1/3) times less number of registers. To get the required SNR at the output of the decimation filter 24 bit register is sufficient. So output CIC section is right shifted by 5 bits which makes the input to Halfband1 is 24 bit data.

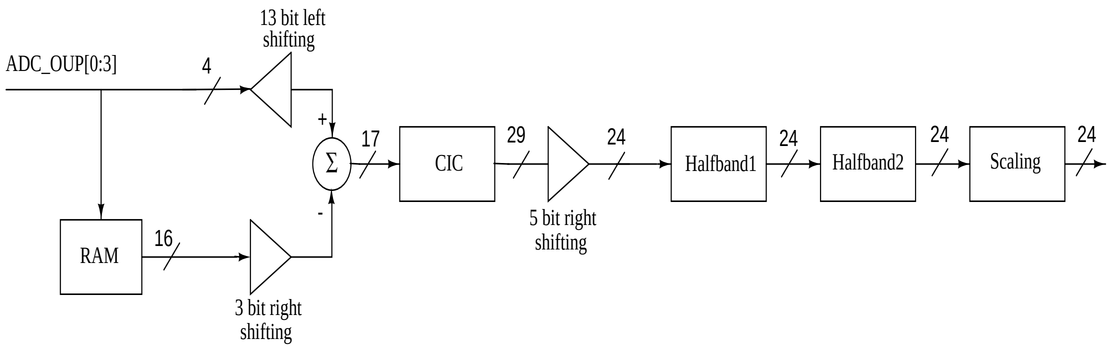


Figure 7.5: Digital correction doen before the decimation filter

The output spectrums of the decimation filter showing performance improvement with digital correction, are shown in Figure 7.6. Figure 7.6(a) and Figure 7.6(b) shows the output spectra for the ideal case and in presence the DAC non-linearities. The latter shows an increased in noise floor and significant harmonic distortion. Figure 7.6(c) shows the output spectrum after obtaining the correction once. The noise level is nearly down to the ideal level. Distortion is reduced significantly, but still visible above the noise floor. Figure 7.6(d) shows the output of the decimation filter after determining the correction values for second time with the improved modulator. The noise and distortion are down to ideal levels. The output SNR and harmonic distortions values after decimation filter are summarized in Table 7.4.

The behavioural model of the decimator structure for digital correction, explained in Figure 7.5 is written in verilog, synthesized with Synopsis Design-Vision and place&routed with Cadence Encounter. Details are summarized in Table 7.3 and power dissipation details for each section are tabulated in Table 7.5.

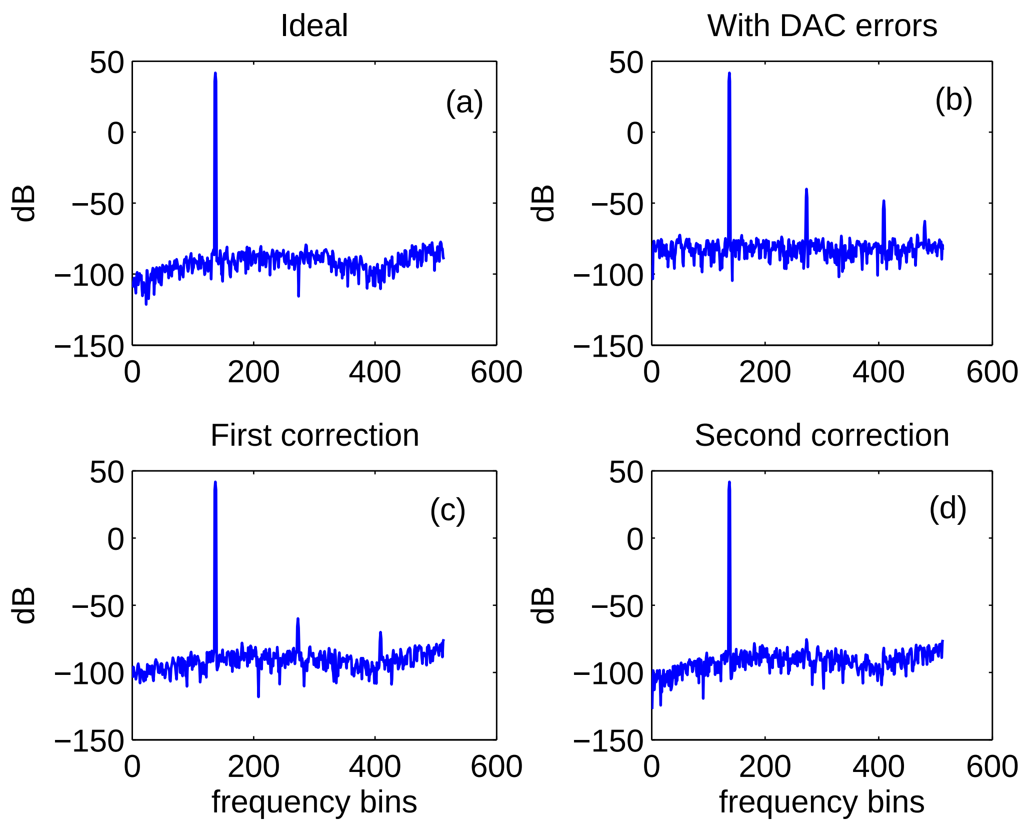


Figure 7.6: Decimator output spectrums showing performance improvement with digital correction.

Table 7.3: Design summary of decimator designed for digital correction

Technology	0.18μm CMOS
Supply Voltage	1.8 V
Total power	75.2 μW
Core area	291753μm^2 ($\approx 546 \times 534$)

Table 7.4: Digital correction performance comparison after decimator

	SNR	HD_2	HD_3
Ideal	104.3 dB	-	-
With DAC errors	81.7 dB	-81.8 dB	-90.0 dB
First correction	99.54 dB	-101.6 dB	-111.8 dB
Second correction	104.1 dB	-117.1 dB	-123.52 dB

Table 7.5: Power report of the decimation filter

Module	**Input Clock**	**Power**	**% Power**
$SINC^4$ filter	1.536 MHz	38.7 μW	51.5
Halfband one	192 KHz	7.53 μW	9
Halfband two	98 KHz	16.2 μ W	21.5
Scaling Block	48 KHz	0.83 μW	1.1
Misc (clock dividers clock buffers)		12.02μW	15.9
Total		75.28 μW	100

7.2 Conclusions

An efficient method for estimating mismatch errors in the feedback DAC for digital correction in $\Delta\Sigma$ A/D converters is proposed and implemented for a 16 level, $OSR = 32$, $OBG = 3$ modulator. This digital correction algorithm requires no reconfiguration of the modulator loop. This technique also does not require switches in signal path that can add to excess loop delay in continuous-time DS modulator. A low power decimator block also designed and implemented to digitally correct the DSM output.

REFERENCES

[1] S. Pavan, N. Krishnapura, R. Pandarinathan, and P. Sankar, “A power optimized continuoous-time $\Delta\Sigma$ ADC for audio applications,” *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 351–360, February 2008.

[2] R. Schreier, *The Delta-Sigma Toolbox Version 7.3*, July 2009.

[3] S. K. Mitra, *Digital Signal Processing*, 3^{rd} ed. Tata McGraw-Hill, 2006.

[4] J. Carol Barrett, “Low-power decimation filter design for multi-standard transceiver applications,” Master’s thesis, University of California, Berkeley.

[5] E. B. Hogenauer, “An economical class of digital filters for decimation and interpolation,” in *IEEE Trans. on Acoust., Speech and Signal Processing*, vol. ASSP-29, no. 2, April 1981, pp. 155–162.

[6] N. Krishnapura, “Efficient Determination of Feedback DAC Errors for Digital Correction in Delta-Sigma A/D Converters,” in *international Symposium on Circuits and Systems (ISCAS), Paris, France*, 31 May-2 Jun. 2010.

[7] S. Norsworthy, R. Schreier, G. Temes, *et al.*, *Delta-sigma data converters: theory, design, and simulation.* IEEE press New York, 1997.

[8] J. Silva, U. Moon, J. Steensgaard, and G. Temes, “Wideband low-distortion delta-sigma adc topology,” *Electronics Letters*, vol. 37, no. 12, pp. 737–738, Jun 2001.

[9] N. R. Doppalapudi, “Decimator, interpolator, and dem techniques for oversampling $\Delta\Sigma$ data converters,” Master’s thesis, IIT Madras, 2006.

[10] J. C. Candy, “Decimation for sigma delta modulation,” in *IEEE Trans. on Communications*, vol. COM-34, no. 1, Jan 1986, pp. 72–76.