A POWER EFFICIENT POLYPHASE SHARPENED CIC DECIMATION FILTER

FOR SIGMA-DELTA ADCs

A Thesis
Presented to
The Graduate Faculty of The University of Akron

In Partial Fulfillment
of the Requirements for the Degree
Master of Science

Nikhil Reddy Karnati
December, 2011
A POWER EFFICIENT POLYPHASE SHARPENED CIC DECIMATION FILTER
FOR SIGMA-DELTA ADCs

Nikhil Reddy Karnati

Thesis

Approved:  Accepted:

Advisor  Dean of the College
Dr. Kye-Shin Lee  Dr. George K. Haritos

Committee Member  Dean of the Graduate School
Dr. Joan E. Carletta  Dr. George R. Newkome

Committee Member
Dr. Robert Veillette

Department Chair
Dr. Alex De Abreu-Garcia

Date
ABSTRACT

The current thesis presents the design and implementation of a power-efficient poly-phase decimation filter for Sigma-delta ADCs. A cascade of Cascaded-Integrator Comb (CIC) filter and Sharpened Cascaded-Integrator Comb (SCIC) filter is used as the decimation filter. This thesis further develops the idea of the two stage architecture by careful consideration of all the possibilities of implementing the two stage structure so as to achieve desired frequency characteristics with minimum power consumption. The architecture of the SCIC stage is slightly modified such that poly-phase decomposition can be easily applied. Therefore, both the CIC and SCIC stages are implemented in poly-phase form, which enables additional power consumption reduction. Various architectures for the poly-phase decomposition of the CIC and SCIC cascaded filter are considered and the architecture that consumes minimum power is chosen. The proposed architecture consumes a power of 1.05 mW and occupies an area of 0.11 mm² using CMOS 1µm technology. The proposed design shows the lowest power consumption among similar filter architecture with equivalent performance.
DEDICATION

Dedicated to my Parents.
ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Kye-Shin Lee for his guidance and support throughout my Master’s program. He never stepped back to help and guide his students and that kept me motivated to do a great job in my research work. It was both an honor and privilege to work with him and shall always be grateful to him.

My special thanks to Dr. Joan Carletta and Dr. Robert Veillette for their invaluable assistance and advice on analog/mixed signal IC design. I enjoyed the hours of technical discussions a lot. I feel extremely privileged to be associated with them.

I would also like to thank Dr. Alex De Abreu-Garcia for supporting me financially during my Master’s program. My special appreciation to Mrs. Gay Boden for her help right from the day I have received admission into the University of Akron.

I take this opportunity to thank my fellow research assistants Pradeep, Dileep, Madhu and Utthej for sharing their experiences in mixed-signal circuit design and layout issues. Finally, I would like to thank my parents and my brother for their unyielding love and encouragement throughout my life.
# TABLE OF CONTENTS

| LIST OF TABLES | ix |
| LIST OF FIGURES | x |

## CHAPTER

### I INTRODUCTION

1.1 Motivation ................................................................. 1

1.2 Goals of the thesis ....................................................... 2

1.3 Thesis organization .......................................................... 3

### II DECIMATION FILTER ARCHITECTURES AND RELATED WORK ............ 4

2.1 Decimation filter ................................................................ 4

2.2 Classification of decimation filters ........................................ 5

2.2.1 Cascaded Integrator-Comb (CIC) filters .................................. 6

2.2.2 Sharpened Cascaded Integrator-Comb (CIC) filters .................... 15

2.3 Previous work related to decimation filters for Sigma-delta ADCs.......... 19

### III DESIGN AND ANALYSIS OF DECIMATION FILTER .................. 25

3.1 Decimation in stages .......................................................... 27
3.2 Case 64_0_0: Decimation of 64 using single non-recursive CIC filter ................. 29
3.3 Case 0_0_64: Decimation of 64 using single sharpened CIC filter ...................... 31
3.4 Case 8_8_0: Decimation of 64 using cascade of CIC and sharpened CIC filters ... 34
  3.4.1 Implementation of the second order CIC filter ........................................ 35
  3.4.2 Implementation of the second order sharpened CIC filter .......................... 35
3.5 Decimation of 4 using equiripple FIR filter .................................................. 37
3.6 Comparison of all the possible filter architectures ......................................... 39

IV DECIMATION FILTER IMPLEMENTATION ......................................................... 41
  4.1 Implementation of $H_2(z)$ using poly-phase decomposition .......................... 41
    4.1.1 Case-I: Cascaded decimation by 2 ....................................................... 42
    4.1.2 Case-II: Decomposing into two sub-filters ............................................ 44
    4.1.3 Case-III: Decomposing into four sub-filters ......................................... 45
    4.1.4 Case-IV: Decomposing into eight sub-filters ......................................... 47
  4.2 Architectures for the implementation of sub-filters ...................................... 48
  4.3 Implementing sharpened $H_1(z)$ using poly-phase decomposition .................. 53

V SIMULATION AND RESULTS ............................................................................... 58
  5.1 Poly-phase structure with a commutative component ..................................... 60
  5.2 Clock-gating .................................................................................................. 61
  5.3 Operand isolation ......................................................................................... 62
  5.4 Implementation of proposed architecture ..................................................... 64
5.4.1 Selection of poly-phase architecture for first stage CIC filter.......................... 64

5.4.2 Implementation of second stage SCIC filter .................................................. 66

5.5 Comparison of architectures............................................................................. 67

VI CONCLUSION AND FUTURE WORK ................................................................ 69

6.1 Summary ............................................................................................................ 69

6.2 Future work ....................................................................................................... 69

BIBLIOGRAPHY ..................................................................................................... 71
LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Possible combinations of decimation factors to implement decimation of 64</td>
<td>29</td>
</tr>
<tr>
<td>3.2 Power estimation of third order non-recursive CIC filter with decimation of 64</td>
<td>32</td>
</tr>
<tr>
<td>3.3 Power estimation of second order sharpened CIC filter with decimation of 64</td>
<td>33</td>
</tr>
<tr>
<td>3.4 Power estimation of $H_2(z)$</td>
<td>37</td>
</tr>
<tr>
<td>3.5 Power estimation of sharpened $H_1(z)$</td>
<td>37</td>
</tr>
<tr>
<td>3.6 Power comparison of all the combination of decimation factors to implement decimation of 64</td>
<td>38</td>
</tr>
<tr>
<td>4.1 Architectures for implementing $H_2(z)$</td>
<td>50</td>
</tr>
<tr>
<td>5.1 Area and power comparison of poly-phase architectures</td>
<td>65</td>
</tr>
<tr>
<td>5.2 Comparison of architectures</td>
<td>67</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Block diagram of decimation filter</td>
<td>5</td>
</tr>
<tr>
<td>2.2</td>
<td>Spectra for decimation factor $N_{tot} = 8$</td>
<td>6</td>
</tr>
<tr>
<td>2.3</td>
<td>Frequency response of CIC filter of order $1$ with $N = 8$</td>
<td>8</td>
</tr>
<tr>
<td>2.4</td>
<td>Frequency response of CIC filter of orders $K = 1$ to $5$ for $N = 8$</td>
<td>8</td>
</tr>
<tr>
<td>2.5</td>
<td>Stopband attenuation of the CIC filter with different values of $K$ and $N$</td>
<td>11</td>
</tr>
<tr>
<td>2.6</td>
<td>Passband droop of the CIC filter with different values of $K$ and $N$</td>
<td>11</td>
</tr>
<tr>
<td>2.7</td>
<td>Block diagram of CIC decimation filter of order $K$ and decimation factor $N$</td>
<td>12</td>
</tr>
<tr>
<td>2.8</td>
<td>Recursive CIC filter of order $K$ and decimation factor $N$</td>
<td>12</td>
</tr>
<tr>
<td>2.9</td>
<td>Block diagram of non-recursive CIC filter of order $K$ and decimation factor $N$</td>
<td>14</td>
</tr>
<tr>
<td>2.10</td>
<td>Non-recursive implementation of CIC filter</td>
<td>14</td>
</tr>
<tr>
<td>2.11</td>
<td>Frequency response of SCIC filter with $K = 1$ to $5$ and $N = 8$</td>
<td>17</td>
</tr>
<tr>
<td>2.12</td>
<td>Passband droop of SCIC filter of different values of $K$ and $N$</td>
<td>18</td>
</tr>
<tr>
<td>2.13</td>
<td>Stopband attenuation of SCIC filter of different values of $K$ and $N$</td>
<td>19</td>
</tr>
<tr>
<td>2.14</td>
<td>Comparison of frequency response of CIC and SCIC filters</td>
<td>21</td>
</tr>
<tr>
<td>2.15</td>
<td>Comparison of passbands of CIC and SCIC filters</td>
<td>21</td>
</tr>
<tr>
<td>2.16</td>
<td>Block diagram of the Sharpened CIC of order-2</td>
<td>22</td>
</tr>
<tr>
<td>2.17</td>
<td>Implementation of Sharpened CIC using basic blocks</td>
<td>23</td>
</tr>
<tr>
<td>2.18</td>
<td>Block diagram of cascade of CIC and SCIC filters</td>
<td>23</td>
</tr>
</tbody>
</table>
2.19 Block diagram of cascade of SCIC filters ......................................................... 24
2.20 Block diagram of cascade of SCIC and FIR filters ........................................... 24
2.21 Block diagram of cascade of CIC and SCIC filters ......................................... 24
3.1 Decimation in stages ............................................................................................ 26
3.2 a) Frequency response of third order CIC filter with decimation factor 64
                  b) Frequency response including equiripple FIR filter .............................. 31
3.3 Block diagram of 3rd order non-recursive CIC filter (decimation of 64) ............ 32
3.4 a) Frequency response of second order sharpened CIC filter
                  b) Frequency response including equiripple FIR filter .............................. 33
3.5 Block diagram of decimation filter ..................................................................... 33
3.6 Block diagram of cascade of non-recursive and sharpened CIC filters ............ 33
3.7 a) Frequency response of cascade of non-recursive and sharpened CIC filters
                  b) Frequency response including equiripple FIR filter .............................. 36
3.8 Non-recursive implementation of $H_2(z)$ ................................................................ 36
3.9 Frequency response of equiripple FIR filter with a decimation of 4 ............... 39
4.1 Poly-phase implementation of $(1 + z^{-1})^2$ using two sub-filters
                  a) Downsampler outside the loop
                  b) Downsampler inside the loop
                  c) Downsampler before the sub-filters .......................................................... 43
4.2 Implementing $H_2(z)$ with two sub-filters ......................................................... 45
4.3 Implementing $H_2(z)$ with four sub-filters ....................................................... 48
4.4 Implementing $H_2(z)$ with eight sub-filters .................................................... 49
4.5 Implementation of $H_2(z)$ in direct form ......................................................... 51
4.6 Implementation of $H_2(z)$ in direct form exploiting symmetry ........................................... 52
4.7 Implementation of $H_2(z)$ in transpose direct form .......................................................... 53
4.8 Implementation of $H_2(z)$ in transpose direct form eliminating delay chains ............... 54
4.9 Block diagram of sharpened $H_1(z)$ .................................................................................. 55
4.10 Implementation of sharpened CIC decimation filter using basic block ......................... 55
4.11 Implementation of the first poly-phase approach ............................................................. 56
4.12 Implementation of the second poly-phase approach ......................................................... 56
5.1 Implementing $H_2(z)$ with eight sub-filters ....................................................................... 59
5.2 Poly-phase implementation of transpose direct form of CIC filter ................................. 59
5.3 Circuit with a multiplier and flip-flop ................................................................................ 61
5.4 Circuit with clock gating ................................................................................................... 61
5.5 Circuit without operand isolation ..................................................................................... 63
5.6 Circuit with operand isolation ......................................................................................... 63
5.7 Block diagram of decimation filter .................................................................................. 63
5.8 Block diagram of second stage SCIC filter ..................................................................... 66
CHAPTER I
INTRODUCTION

1.1 Motivation

With the development of more compact and efficient ways of implementing digital logic on a silicon chip, most of the signal processing is being performed in digital domain. The sigma-delta analog-to-digital converter (ADC) is one such application where this trend can be observed. Sigma-delta ADCs provide low cost and low power consumption while achieving higher resolution compared to other ADC architectures. Furthermore, the sigma-delta ADC is used in variety of applications such as communications, signal processing, data acquisition and instrumentation. The sigma-delta ADC comprises a modulator and a decimation filter. Cascaded Integrator-Comb (CIC) filters are used as decimation filter. However, CIC filters have a monotonic passband characteristic with significant passband droop. Therefore, an additional droop compensation filter is required. The sharpened CIC (SCIC) filters eliminate the need for an additional droop compensation filter [4]. SCIC filters exhibit negligible passband droop and higher stopband attenuation compared to CIC filters of the same order and decimation factor. Although SCIC filters eliminate the need for droop compensation filters, they consume a lot of power. This work focuses on developing a low power sharpened CIC filter.
1.2 Goals of the thesis

The current thesis focuses on implementing a power efficient poly-phase sharpened CIC filter for sigma-delta ADCs. The decimation filter should achieve a SNR of 98 dB which is equivalent to a 16-bit resolution. The input used to test the decimation filter is a bit-stream obtained by applying a 200 kHz sine wave input to a second-order sigma-delta modulator, which corresponds to the bandwidth of the ADC. The sigma-delta modulator operates at an oversampling frequency of 102.4 MHz. So, the decimation filter should decimate the input bit-stream operating at 102.4 MHz to a Nyquist rate of 400 kHz ($2 \times 200$ kHz) with the minimum power consumption.

In this thesis, we first consider the conventional CIC and SCIC decimation filters that achieve the desired SNR. Then, a cascade of CIC and SCIC filters is considered, and the optimum decimation factor of each stage that gives the minimum power consumption achieving the desired SNR is found. Various architectures for poly-phase implementation of the CIC and SCIC filters are considered and the architecture that consumes minimum power is chosen. The SCIC filter transfer function is modified without degrading the filter characteristics such that poly-phase decomposition can be applied to the modified transfer function. The poly-phase implementation not only reduces the frequency at which the filter operates but also eliminates the need for large bit-width integrators and comb sections running at high sample rate. Because the operating frequency and register bit widths are greatly reduced, the power and area requirements of the decimation filter are tremendously reduced compared to all other conventional filter architectures that are published.
1.3 Thesis organization

The rest of the thesis is divided into five chapters. Chapter II presents the conventional CIC and SCIC decimation filters. Previous work on decimation filter architectures is summarized and compared to the proposed architecture. In Chapter III, optimization of the decimation factor for implementing the cascade of CIC and SCIC filters is discussed. In Chapter IV, various architectures for the poly-phase decomposition of the CIC and SCIC cascaded filter are considered and the architecture that consumes minimum power is chosen. In Chapter V, implementation of the proposed architecture is described, followed by the simulation results. In Chapter VI, the results are summarized and possible future work is discussed.
CHAPTER II

DECIMATION FILTER ARCHITECTURES AND RELATED WORK

2.1 Decimation filter

Decimation is the process of lowering the word rate of digitally encoded signals that have been oversampled for above the Nyquist rate. The output of the sigma-delta modulator is a single bit data stream combining two components: the desired signal and the modulator quantization noise. The decimation filter that follows the modulator is required to remove the out-of-band noise and to reduce the sample rate by a factor of $N_{\text{tot}}$, where $N_{\text{tot}}$ is the oversampling ratio of the modulator and also the decimation factor of the decimation filter. The decimation filter is a combination of a low pass filter and a downsampler. A block diagram of the decimation filter is shown in Figure 2.1. The input to the decimator is the 1-bit data stream coming from the modulator. The input to decimator has a high word rate of $F_s$ (sampling frequency of the modulator) and the decimation filter has an output word rate of $F_s/N_{\text{tot}}$. The input signal $x(n)$ is passed through the low pass filter to obtain $w(n)$ and then passed through the downsampler. Figure 2.2 illustrates how the input signal $x(n)$ is transformed to the decimated output signal $y_{\text{out}}(u)$ [3]. Spectra of all the signals $x(n)$, $w(n)$ and $y_{\text{out}}(u)$ are shown, assuming the decimation factor of 8. $|W(f)|$ is obtained by passing $|X(f)|$ through the low pass filter. The input spectrum $|X(f)|$ repeats every $F_s$ and the output spectrum $|Y_{\text{out}}(f)|$ repeats every $F_s/8$. $\Delta F$ is the transition bandwidth (stopband cutoff frequency minus passband cutoff frequency) of the decimation filter.
As large oversampling ratios are required in sigma-delta ADCs, the ratio of sampling frequency to the transition bandwidth $\Delta F$ of the decimation filter tends to be large. FIR filters can be used as decimation filters. The number of taps in an FIR filter is directly proportional to the sampling frequency $F_s$ and inversely proportional to the transition bandwidth $\Delta F$ [3]. As this ratio is usually large for sigma-delta ADCs, the number of taps in conventional FIR filters tends to be high. In order to avoid large number of taps in FIR filters, special FIR filter architectures are used as decimation filter. The following sections discuss various widely used architectures of decimation filters.

2.2 Classification of decimation filters

Decimation filters can be broadly classified into two categories: Cascaded-Integrator-Comb (CIC) filters and Sharpened Cascaded-Integrator-Comb (SCIC) filters. CIC filters are linear-phase low pass FIR filters. These filters exhibit $\sin(N_{tot} \times x)/\sin(x)$ amplitude characteristic. Therefore they are also called sinc filters. As the name suggests, CIC filters can be implemented as a cascade of integrator and comb sections. Sharpened CIC filters are obtained by applying Kaiser and Hamming sharpening technique [5] to the CIC filter. SCIC filters have sharpened frequency response compared to conventional CIC filters. They have better passband and stopband characteristics than CIC filters. Frequency response characteristics and implementation examples of CIC and SCIC filters are explained in the following sub-sections.

![Figure 2.1 Block diagram of the decimation filter](image-url)
Figure 2.2 Spectra for decimation factor $N_{\text{tot}} = 8$

2.2.1 Cascaded Integrator-Comb (CIC) Filters

The simplest low pass FIR filter is the simple moving average filter which has uniform weighting. The impulse response of a simple moving average filter is a rectangular pulse as given by

$$h(n) = \begin{cases} \frac{1}{N} & \text{for } 0 \leq n \leq N-1 \\ 0 & \text{otherwise} \end{cases},$$  \hspace{1cm} (2.1)

where $N$ is the decimation factor. The z-domain transfer function of the simple moving average filter is given by

$$H(z) = \frac{1}{N} \sum_{i=0}^{N-1} z^{-i},$$  \hspace{1cm} (2.2)

which may be modified to obtain transfer function of CIC filter as

$$H(z) = \frac{1}{N} \left( \frac{1-z^{-N}}{1-z^{-1}} \right),$$  \hspace{1cm} (2.3)
The frequency response $H(e^{j\omega})$ following from Equation (2.3) is given by

$$H(e^{j\omega}) = \frac{1}{N} \left( \frac{\sin\left(\frac{\omega N}{2}\right)}{\sin\left(\frac{\omega}{2}\right)} \right) e^{-j\omega(N-1)/2} \quad (2.4)$$

It can be observed from Equation (2.4) that the CIC filter is a linear phase filter which exhibits $\sin(Nx)/\sin(x)$ amplitude characteristic. Figure 2.3 illustrates the frequency response of the CIC filter. It has $N/2$ spectral zeros if $N$ is even and $(N-1)/2$ spectral zeros if $N$ is odd, along the frequency axis at integer multiples of $F_s/N$. The spectral zeros provide very high attenuation at these frequencies. CIC filters have wide transition bands but the passband of interest $f_b$ is usually a small portion of the passband. So they are usually followed by a low pass filter that provides the desired passband $f_b$ and removes the noise in the wide transition band of the CIC filter. Due to insufficient attenuation in the noise at the frequencies around integer multiples of $F_s/N$, they get aliased into the desired passband. These are called aliasing bands [4]. These aliasing bands are found at frequencies

$$f = n \times \left(\frac{F_s}{N}\right) \pm f_b \quad n = \{1, 2, 3 \ldots N/2\}. \quad (2.5)$$

For $N = 8$, there are four aliasing bands. These aliasing bands are labeled as $\phi$ in Figure 2.3. The aliasing bandwidths near the spectral zeros are too narrow to provide sufficient attenuation of aliasing noise in the entire baseband signal.

The stopband attenuation is only 12.8dB for a CIC filter of first order with $N = 8$. It can be improved by using CIC filters of higher order. The transfer function of a CIC filter of order $K$ is given by
$$H(z) = \left[ \frac{1}{N} \left( 1 - z^{-N} \right) \right]^K.$$ (2.6)

Figure 2.3 Frequency response of CIC filter of order = 1 with $N = 8$

Figure 2.4 Frequency response of CIC filter of orders $K = 1$ to 5 for $N = 8$
The higher order filters have multiple spectral zeros with multiplicity equal to the order of the filter. Due to the presence of these multiple spectral zeros, higher order CIC filters exhibit greater stopband attenuation. Figure 2.4 illustrates the frequency response of CIC filters of order $K = 1$ to $5$. It can be inferred from the plots that the stopband attenuation increases with the increase in order of the filter. A CIC filter of order five achieves stopband attenuation of $64.21\text{dB}$ with $N = 8$. Figure 2.5 depicts the stopband attenuation of the CIC filter with various orders and different decimation factors. CIC filters of higher order exhibit better stopband characteristic. Stopband attenuation remains constant even with the increase in decimation factor.

However, it can also be observed from Figure 2.4 that there is a significant droop in the passband of the CIC filter. The passband droop is the attenuation in frequency characteristic of the filter in the passband. Ideally the passband droop should be zero. The passband droop becomes worse as the filter order increases. The passband droop also increases with the increase in decimation factor $N$. The rate of increase in the passband droop is larger for higher decimation factors. Figure 2.6 depicts the passband droop of CIC filter of various orders with different decimation factors. Width of the desired passband $f_b$ is taken as $F_s/512$. The passband droop has to be compensated in the next stage filter that follows the CIC filter. For a system that requires programmable decimation factors, the next stage compensation filter has to be a programmable filter; such filters occupy a large amount of area and consume a lot of power [6].

CIC filters are classified into recursive CIC filters and non-recursive CIC filters, depending on the architecture. Recursive CIC filters include recursive structures such as integrators, whereas non-recursive CIC filters do not include any integrators, and instead
make use of non-recursive blocks for implementing the filter. The following subsections compare the two architectures in detail.

(A) Recursive CIC filters

Hogenauer proposed an architecture that uses both recursive and non-recursive blocks for CIC filter implementation [4]. A CIC filter of order $K$ is implemented as a cascade of $K$ integrator sections and $K$ comb sections. A block diagram of the CIC decimation filter is shown in Figure 2.7. Hogenauer further proposed to move the downsampler block before the comb section. Therefore, the integrator section runs at the input data rate $F_s$ and the comb section runs at the reduced sampling rate $F_s/N$, as shown in Figure 2.8. Although register overflow may occur at each integrator stage as each stage has a unity feedback, the register overflow is of no consequence if the number system used is two’s complement arithmetic, or another number system that allows wrap-around between most positive and most negative numbers [4].

In addition, the maximum range of the number system should be equal to or exceed the maximum magnitude expected at the output of composite filter [4]. The maximum number of bit required at the output of the filter is calculated to be

$$B_{\text{max}} = K \times \log_2 N + B_{\text{in}}$$

where $B_{\text{in}}$ is the number of bits of the input signal [4]. As a result, the word-length of all the integrator stages has to be greater than or equal to $B_{\text{max}}$ to ensure that no data is lost due to register overflows. $B_{\text{max}}$ is the significant and the highest order bit in each integrator section as seen at the output register. Since a propagation path has to be provided through the comb sections, $B_{\text{max}}$ is also the MSB bit for the comb sections that
follow. Therefore, all the stages, both integrator and comb sections, should have a word-length of $B_{\text{max}}$ to avoid the overflow.

Figure 2.5 Stopband attenuation of the CIC filter with different values of $K$ and $N$

Figure 2.6 Passband droop of the CIC filter with different values of $K$ and $N$
Although recursive CIC filters do not need multipliers for implementation, they suffer from several drawbacks. First, the monotonic passband characteristic of a CIC filter produces an inevitable passband droop that has to be compensated in the second-stage filter. Second, the frequency response can be controlled by only two parameters, filter order $K$ and decimation factor $N$, and so only a limited range of filter characteristics can be obtained. Third, register-width becomes large for large decimation factor $N$, and fourth, the integrator sections operate at maximum sampling rate $F_s$ which leads to very high power consumption and large area. The circuit speed is also limited by the large word-length and recursive loop of the integrator section.

(B) Non-recursive CIC filters

As the recursive CIC filters require large word-length integrators operating at maximum sampling rate, there was a need for developing a structure which eliminates the need for such integrators. This has given rise to the non-recursive CIC filter, which does not require recursive blocks [19]. The conventional CIC transfer function is equivalent to a summation of delay elements with unity coefficients, as

$$H(z) = \left[ \frac{1-z^{-N}}{1-z^{-1}} \right]^K = \left[ \sum_{i=0}^{N-1} z^{-i} \right]^K.$$

(2.8)
This is further modified into cascade of non-recursive blocks by making use of the commutative rule, as

\[ H(z) = \left[ \sum_{i=0}^{N-1} z^{-i} \right]^K = \left[ \sum_{i=0}^{2^{M-1}-1} z^{-i} \right]^K = \prod_{i=0}^{M-1} (1 + z^{-2^i})^K \]  \hspace{1cm} (2.9)

where

\[ \prod_{i=0}^{M-1} (1 + z^{-2^i})^K = (1 + z^{-1})^K (1 + z^{-2})^K (1 + z^{-4})^K (1 + z^{-8})^K \ldots (1 + z^{-2^{M-1}})^K \]  \hspace{1cm} (2.10)

Finally, it can be observed from Equation (2.9) that non-recursive architecture requires \( M \) sections where

\[ M = \log_2 N \]  \hspace{1cm} (2.11)

For a filter of order \( K \), each section has \( K \) copies of non-recursive blocks. Figure 2.9 illustrates the implementation of a CIC decimation filter as a cascade of non-recursive blocks. The downsampler (factor of \( 2^M \)) block can be distributed through the sections, so that each section works at a sampling rate less than the one before it. Figure 2.10 illustrates the implementation of CIC filter after distributing the downsampler so that the sampling rate is reduced after each section by a factor of 2.

Since the order of each section is \( K \), the word-length increases through each section by only \( K \) bits. Though section-1 runs at input data rate \( F_s \), it has word-length of only \( K + B_{in} \) bits. The word-length of section-2 is \( 2K + B_{in} \) bits, and it runs at a sampling rate of \( F_s/2 \). The word-length of the last section, section-\( M \), is

\[ B_{max} = KM + B_m = K \log_2 N + B_m \]  \hspace{1cm} (2.12)
Figure 2.9 Block diagram of non-recursive CIC filter of order $K$ and decimation factor $N$

Figure 2.10 Non-recursive implementation of CIC filter

The last section runs at frequency $F_s/2^{M-1}$. The word-length is short when the sampling rate is high and the word-length increases as the sampling rate decreases. Therefore, the power consumption is reduced when compared with a recursive implementation, which has a long word-length and a high sample rate throughput. Circuit speed is also increased because the first section in a non-recursive structure always has a smaller word-length than the integrator stage in a recursive structure. Non-recursive CIC filters have several advantages over recursive CIC filters. Large word-length integrators operating at maximum sampling rate are eliminated in this architecture. Register word-length increases through each section by only $K$ bits, independent of the value of the decimation factor $N$. Therefore, large values of $N$ do not affect the word-length. Only the final section in the non-recursive structure has the maximum word-length $B_{\text{max}}$ while all the integrators in the recursive structure have that maximum word-length. All the above characteristics result in low power consumption. The circuit speed is also improved as it is not limited by large word-length and recursive loop of the integrator section which was a drawback for recursive architectures. But the non-recursive CIC filters still suffer from several drawbacks. Since both the recursive and the non-recursive CIC filters have the
same frequency characteristics, the non-recursive architectures also have significant passband droop in the frequency response. The issue of passband droop which has to be compensated in the second-stage filter is not yet addressed. Similar to recursive architectures, the frequency response of non-recursive architecture is also fully controlled by only two parameters, namely the filter order $K$ and the decimation factor $N$, which results in limited range of filter characteristics. The frequency at which each stage operates can be reduced to further minimize power consumption.

2.2.2 Sharpened Cascaded-Integrator-Comb (SCIC) filters

CIC filters of higher order are required to achieve better passband and stopband characteristics. However, higher order CIC filters not only suffer from significant passband droop but also consume a lot of power and occupy large area. The passband droop of conventional CIC based architecture has to be compensated by a second stage compensation filter. CIC based architecture have variable passband droop when used with multiple decimation factors. The limitation of the CIC-based architecture for programmable decimating digital filters are due to the need for a programmable second-stage filter to compensate for the variable droop in the passband of the CIC filter response. These programmable filters require large area and consume a lot of power. Sharpened CIC filters not only achieve better aliasing rejection and stopband attenuation but also exhibit negligible passband droop [6] and eliminate the need for programmable second-stage filter.

Kaiser and Hamming propose a method to sharpen the magnitude response of a digital filter by using multiple realizations of low-order basic filters [5]. This filter
sharpening technique is applied to CIC filters to reduce the passband droop and improve stopband attenuation in CIC filters. A family of sharpening filters $H_{nm}(f)$ is given by

$$H_{nm}(f) = H^{n+m}(f) \sum_{k=0}^{m} \frac{(n+k)!}{n!k!} [1-H(f)]^k$$  \hspace{1cm} (2.13)

where $H(f)$ is the low-order basic filter. By choosing integers $n$ and $m$, we select order of tangency at frequencies where $H_{11}(f) = 0$ and $H_{11}(f) = 1$ [5]. Choosing $n = m = 1$,

$$H_{11}(f) = H^{1+1}(f) \sum_{k=0}^{1} \frac{(1+k)!}{1!k!} [1-H(f)]^k$$  \hspace{1cm} (2.14)

$$H_{11}(f) = H^2(f)[3-2H(f)]$$  \hspace{1cm} (2.15)

Assuming the group delay of $H(z)$ to be $D$ samples, where $D = \left\lfloor \frac{N-1}{2} \right\rfloor \times K$,

$$H_{11}(z) = [H(z)]^2[3z^{-D} - 2[H(z)]]$$  \hspace{1cm} (2.16)

Here the term $z^{-D}$ is introduced to make the group delays uniform. The sharpened CIC filter is obtained by replacing the basic filter $H(z)$ with the CIC filter transfer function

$$H_{SCIC}(z) = \left[ \frac{1}{N} \left( \frac{1-z^{-N}}{1-z^{-1}} \right) \right]^{2K} \left[ 3z^{-D} - 2 \left( \frac{1-z^{-N}}{1-z^{-1}} \right)^K \right].$$  \hspace{1cm} (2.17)

The magnitude response of the SCIC filter with order $K$ is given by

$$|H_{SCIC}(e^{j\omega})| = \left| 3 \left( \frac{1}{N} \frac{\sin(\omega N/2)}{\sin(\omega/2)} \right)^{2K} - 2 \left( \frac{1}{N} \frac{\sin(\omega N/2)}{\sin(\omega/2)} \right)^{3K} \right|$$.  \hspace{1cm} (2.18)

Figure 2.11 illustrates the frequency response of SCIC filters of order $K = 1$ to 5. It can be inferred from the plots that the stopband attenuation increases as the order of the filter increases without much increase in the passband droop, which is almost negligible. The
fifth order SCIC filter achieves stopband attenuation of 118.4dB with \( N = 8 \). Figure 2.12 depicts the passband droop and Figure 2.13 plots the stopband attenuation of SCIC filters of various orders with different decimation factors [6]. The passband droop is almost zero even with the increase in decimation factor and the order of the filter. Maximum passband droop of 0.36dB is observed for the fifth order with decimation factor of 64.

The stopband attenuation increases tremendously with the increase in order of the filter and it is constant with the increase in decimation factor. Figure 2.14 compares the frequency response of the conventional CIC filter and the SCIC filter of order \( K = 2 \) and decimation factor \( N = 64 \). Figure 2.15 shows the comparison of passband responses of the CIC and SCIC filters. It can be observed from the plots that SCIC filter has sharpened filter response with negligible passband droop and far better stopband attenuation than the conventional CIC filter. The aliasing bandwidths of SCIC filter are wider than CIC filters and provide sufficient attenuation of aliasing in the entire signal baseband.
The block diagram of the SCIC filter of order 2 is shown in Figure 2.16. It can be observed that the implementation requires three identical CIC filters, two multipliers of values -2 and 3, and an adder.

Figure 2.17 illustrates the implementation of a second order sharpened CIC decimation filter using basic building blocks [6]. It requires six integrator sections, six comb sections, two multipliers, one adder and a delay block. All the integrators are placed before the downsampler. Therefore, they operate at a high sampling rate. All the comb sections are placed after the downsampler so that they operate at reduced sampling rate. This implementation suffers from the same drawbacks as the recursive CIC filters shown in Figure 2.8. That is, large word-length integrators operating at high sampling rates are required in this implementation. It is these integrators that reduce the circuit speed, consume a large amount of power, and occupy a large area. The comb sections operate at lower frequency so they consume less power than the integrators. Therefore,
the primary area of concern is to eliminate these large word-length integrators. In this thesis, a novel architecture that reduces the sampling rate while eliminating the large word-length integrators is proposed.

2.3 Previous work related to decimation filters for sigma-delta ADCs

Low power design of sigma-delta ADCs that achieves high bit-resolution is the need of the hour. So far, although significant work has been done on a wide range of sigma-delta modulators, from multi-order modulators to multi-channel modulators, not much work has been done on decimation filters for sigma-delta ADCs. Decimation filters play an equally important role in designing low power sigma-delta ADCs. The techniques that are developed to design a low power decimation filter for sigma-delta ADC are discussed in this section.

Figure 2.13 Stopband attenuation of SCIC filter of different values of $K$ and $N$
Gao and Jia [12] proposed a non-recursive algorithm for implementing the CIC filter. It uses a non-recursive structure that is described in Section 2.2.1 (B). Nerurkar and Abed proposed a decimation filter which uses a cascade of non-recursive CIC, IIR and half-band FIR filters [20]. The use of IIR filters makes the architecture highly sensitive to pole location, and related stability issues are a limitation. Though the non-recursive CIC filter eliminates the need for integrators, the first stage filter still runs at a high sampling rate. In addition, the passband droop requires an additional compensation filter. The aliasing bandwidths near the spectral zeros are too narrow to provide sufficient attenuation of aliasing noise in the entire baseband signal. High order filters that consume large power and occupy a large area are required to achieve the design specifications. The filter architecture proposed in this thesis consumes less power to achieve the same design specification as the conventional filter. Detailed performance comparison of the proposed architecture with the non-recursive CIC filter architecture is shown in the following chapter.

Kwentus, Jiang and Willson [6] developed a sharpened CIC filter that addresses the issue of passband droop in CIC filters. G.J. Dolecek and S.K. Mitra proposed the cascade of CIC and SCIC filters [7] shown in Figure 2.18. Both of them have implemented a sharpened CIC filter as a cascade of integrator and comb sections as shown in Figure 2.17. This implementation suffers from the same drawbacks as the recursive CIC filters. Large word-length integrators are needed in this implementation. As the most significant bit has to be propagated to the output of the filter, the comb sections also require large word-length registers as integrators. The result is an implementation that requires large area and consumes significant power.
Figure 2.14 Comparison of frequency response of CIC and SCIC filters

Figure 2.15 Comparison of passbands of CIC and SCIC filters
Figure 2.16 Block diagram of the Sharpened CIC of order-2

Rajic and Babic [9] proposed a method of splitting the total decimation factor into lower decimation factors and implementing the system as a cascade of SCIC filters as shown in Figure 2.19. Cascading SCIC filters achieves better frequency response but results in a lot of hardware and greater power consumption. Liu and Willson [8] used a cascade of a single SCIC (decimation of 4) and three half-band FIR filters (decimation of 2 each) to realize a decimation of 32 as shown in Figure 2.20. However, the design is not good for higher decimation factors as it requires an SCIC filter with higher decimation factor and large word-length registers. Some filter architectures [28] used a compensation filter in addition to the SCIC filter to improve the frequency characteristics at the expense of more hardware. G.J. Dolecek and S.K. Mitra [10] used a single CIC filter with a decimation factor of 2 before the SCIC filter. G. Stephen and R.W. Stewart [11] used a cascade of three CIC filters (with decimation factor 2 each) before a SCIC filter as shown in Figure 2.21. Both these architectures limit themselves in that they perform decimation of only 2 in the initial stages. Higher power saving can be achieved using CIC filters performing decimation of more than 2 in the initial stages. The other filter architectures used higher order amplitude change function for filter sharpening [2]. This would improve the frequency characteristics but at the cost of large power and hardware.
The architecture proposed in this thesis uses a novel architecture for the implementation of sharpened CIC filter. The SCIC filter transfer function is modified without degrading the filter characteristics such that poly-phase decomposition can be applied to the modified transfer function. Poly-phase implementation greatly reduces the frequency at which the filter operates. In addition, the registers that are present at the input do not require large word-length. This architecture thus eliminates the need for large word-length integrators and comb sections running at high sampling rate. Furthermore, since the SCIC filter is not placed at the input side, which is followed by other decimation blocks, the sampling rate of the SCIC filter can be lowered. Therefore by greatly reducing the operating frequency and register word-length, the power and area requirements of the decimation filter are tremendously reduced compared to all other conventional filter architectures that are published.

Figure 2.17 Implementation of Sharpened CIC using basic blocks

Figure 2.18 Block diagram of cascade of CIC and SCIC filters [7]
Figure 2.19 Block diagram of cascade of SCIC filters [9]

Figure 2.20 Block diagram of cascade of SCIC and FIR filters [8]

Figure 2.21 Block diagram of cascade of CIC and SCIC filters [11]
CHAPTER III
DESIGN AND ANALYSIS OF DECIMATION FILTER

The decimation filter that follows the sigma-delta modulator should achieve an SNR of 98.08dB, which is equivalent to a 16-bit resolution. The input of the decimation filter is a bit-stream obtained by applying a 200 kHz sine wave input to a second-order sigma-delta modulator. This frequency corresponds to the bandwidth of the ADC. The sigma-delta modulator operates at an oversampling frequency of 102.4MHz. So, the decimation filter should decimate the input bit-stream operating at 102.4MHz to a Nyquist rate of 400 kHz (2 × 200 kHz) with the minimum power consumption. The total decimation factor to be achieved is

\[ N_{tot} = \frac{102.4 \text{ MHz}}{400 \text{ kHz}} = 256. \]  

The decimation filter can be implemented either in a single stage or in multiple stages. As mentioned in Chapter II, the number of taps required for an FIR decimation filter is directly proportional to the ratio of the sampling frequency to the transition bandwidth [3]. Consider a single stage FIR filter with a decimation factor of 64 which converts a signal from frequency \( F_s \) to \( F_s/64 \) with a baseband extending from zero to \( F_s/128 \). The transition band of the filter can extend from \( F_s/128 \) to \( F_s/128 + \Delta F \), where \( \Delta F \) is a very small value compared to the sampling frequency. For example, sampling frequency \( F_s \) can be in 102.4 MHz and \( \Delta F \) can be 10 kHz. In this case, the ratio of the
sampling frequency to transition bandwidth is large. Therefore, a large number of taps is required to realize this FIR filter.

Now consider a two-stage implementation in which the first stage and the second stage FIR filters perform decimations of 32 and 2, respectively. The first stage FIR filter reduces the sampling frequency from $F_s$ to $F_s/32$, and the noise which is present in its transition band is attenuated in the second stage FIR filter. Therefore, the transition region of the first stage FIR filter can be made wide enough so that the transition bandwidth of the filter is not small compared with its sampling frequency; as a result, a lower number of taps is required to implement the first stage FIR filter. The second stage FIR filter has the same transition band as that of the FIR filter in a single stage (i.e. $F_s/128$ to $F_s/128 + \Delta F$) but referenced to a much lower sampling frequency $F_s/32$. Therefore the total number of taps required in the two-stage filter is small compared to the number required in a single-stage filter. A design example illustrated in [3] demonstrates the advantages of multi-stage decimation filtering.

![Figure 3.1 Decimation in stages](image-url)

26
3.1 Decimation in stages

Decimation filters can be implemented in an arbitrary number of stages. In addition, the total decimation factor $N_{\text{tot}}$ can be split into various combinations. The efficiency of the design mainly depends on the number of stages, the amount of decimation that each stage performs, and the type of filter used in each stage. It is shown that the CIC filter can efficiently decimate the output of sigma delta modulator to four times the Nyquist rate [3].

Therefore the decimation of $N_{\text{tot}}$ of the overall digital filter is first split into $N_c$ and 4 such that $N_{\text{tot}} = N_c \times 4$ where decimation of $N_c$ is performed in the CIC filter, and decimation of 4 is done in an FIR filter. The decimation of $N_c$ can be implemented using different kinds of CIC filters as mentioned in Chapter II. Decimation factor $N_c$ can also be implemented in multiple stages. The particular decimation filter designed in this chapter has $N_{\text{tot}} = 256$, with CIC filter performing decimation of 64. The equiripple FIR filter described in Section 3.5 is used to provide the additional decimation of 4 so that the total decimation of 256 is achieved. For all cases, the same equiripple FIR filter is used as the final FIR filter stage.

The next step is to determine all the combinations of decimation factors and types of filters that can be used to realize the CIC filter with a decimation factor of 64. The focus is on finding the lowest power solution. The proposed architecture is compared with other conventional architectures in the following sections. All the combinations of realizing the decimation factor in one, two and three stages are compared. Realizing the decimation factor $N_c$ in four or more stages would add extra sharpening stages, which means more hardware and power. First, the decimation factor $N_c = 64$ is divided into $N1$,
$N_2$ and $N_3$ such that $N_c = N_1 \times N_2 \times N_3$ where $N_1$, $N_2$ and $N_3$ are performed by the first, second and third stages respectively. The first stage is realized with a conventional non-recursive CIC filter since non-recursive CIC filters have lower power consumption than recursive CIC filters [12]. Sharpened CIC filters are used for the second and third stages. Table 3.1 shows all the possible ways in which $N_c = 64$ can be split into three factors. For instance, Case 1_1_64 has $N_1 = N_2 = 1$ and $N_3 = 64$, which indicates that the overall implementation consists of a single sharpened CIC filter stage performing the decimation of 64. Case 2_2_16 has a non-recursive CIC filter that provides a decimation of $N_1 = 2$ and two sharpened CIC filters that provide decimations of $N_2 = 2$ and $N_3 = 16$, respectively. Figure 3.1 illustrates the concept of decimation in stages.

A power model as described in [22] is used to estimate power consumption for each case. It is assumed that power is directly proportional to order of the filter, operating frequency of the filter and the word-length as

$$\text{Power} \propto \sum \left( K \times B_{\text{max}} \times F_r \right) \quad (3.2)$$

This power model is used to compare the power consumption of all the possible combinations. However this model is only used as a design guideline. The actual power consumption values, measured from circuit level simulation, are presented in Chapter 4. A number of different architectures for implementing the filter stage of decimation factor 64 are discussed in the following sections. Case 64_1_1 and Case 1_1_64 are described first. Then, Case 8_8_1 which meets the desired specifications with the lowest power solution is deliberated.
Table 3.1 Possible combinations of decimation factors to implement decimation of 64

<table>
<thead>
<tr>
<th></th>
<th>N1</th>
<th>N2</th>
<th>N3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case 1_1_64</td>
<td>1</td>
<td>1</td>
<td>64</td>
</tr>
<tr>
<td>Case 1_2_32</td>
<td>1</td>
<td>2</td>
<td>32</td>
</tr>
<tr>
<td>Case 1_4_16</td>
<td>1</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>Case 1_8_8</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Case 1_16_4</td>
<td>1</td>
<td>16</td>
<td>4</td>
</tr>
<tr>
<td>Case 1_32_2</td>
<td>1</td>
<td>32</td>
<td>2</td>
</tr>
<tr>
<td>Case 2_2_16</td>
<td>2</td>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>Case 2_4_8</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Case 2_8_4</td>
<td>2</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>Case 2_16_2</td>
<td>2</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>Case 2_32_1</td>
<td>2</td>
<td>32</td>
<td>1</td>
</tr>
<tr>
<td>Case 4_2_8</td>
<td>4</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Case 4_4_4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Case 4_8_2</td>
<td>4</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>Case 4_16_1</td>
<td>4</td>
<td>16</td>
<td>1</td>
</tr>
<tr>
<td>Case 8_2_4</td>
<td>8</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Case 8_4_2</td>
<td>8</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>Case 8_8_1</td>
<td>8</td>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>Case 16_2_2</td>
<td>16</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Case 16_4_1</td>
<td>16</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Case 32_2_1</td>
<td>32</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Case 64_1_1</td>
<td>64</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

3.2 Case 64_1_1: Decimation of 64 using single non-recursive CIC filter

Case 64_1_1 implements the decimation filter as single stage non-recursive CIC filter with decimation of 64, followed by the equiripple FIR filter with a decimation of 4.

The first step in designing a non-recursive CIC filter is to determine the order of the filter.
that provides the desired passband and stopband characteristics. These characteristics determine the signal-to-noise ratio (SNR) of the ADC.

The order of the non-recursive CIC filter is chosen such that cascade of non-recursive CIC filter with decimation factor of 64 and equiripple FIR filter with decimation factor of 4 achieves a SNR greater than 98.08dB, which is equivalent to a 16-bit resolution. For a delta-sigma modulator of order \( L \), a CIC filter of order \( K = L + 1 \) is required to adequately attenuate the quantization noise that would alias into the desired signal band [4].

As a result, a third-order CIC filter is required to decimate the output of a second order sigma-delta modulator without loss of the SNR. The frequency response of third order CIC filter is shown in Figure 3.2.a. Third order CIC filter exhibits a stopband attenuation of 39.77dB and a passband droop of 0.673dB. Frequency response including equiripple FIR filter is shown in Figure 3.2.b. This architecture achieves the desired SNR specification of the sigma-delta ADC; however, this configuration requires an additional compensation filter or it should be included in the equiripple FIR filter to compensate the passband droop of CIC filter.

Figure 3.3 shows the non-recursive CIC filter implementation. There are 6 sections and each section has 3 copies of non-recursive blocks. Section-1 runs at frequency of \( F_s \), whereas the operating frequency of each section reduces by 2. The word-length increases through each section by 3 bits since this is a third order filter. Table 3.2 shows the power estimation of third order non-recursive CIC filter with decimation factor of 64, estimated using the power model (Equation 3.2). Again, these numbers are only the power metrics and not the actual power consumption values.
3.3 Case 1_1_64: Decimation of 64 using single sharpened CIC filter

In case 1_1_64, a single sharpened CIC filter is used to provide a decimation of 64. It is observed that a second order sharpened CIC filter provides sufficient attenuation to prevent aliasing of the quantization noise into the signal band. Frequency response of second order sharpened CIC filter is shown in Figure 3.4.a. Figure 3.4.b depicts the frequency response for the complete decimator, which includes the sharpened CIC filter and the equiripple FIR filter. As shown, the filter has negligible passband droop of 0.064dB and high stopband attenuation of 43.76dB, and provides SNR of 100.24dB, which meets the desired SNR specification.

The second order sharpened CIC filter with decimation factor 64 is implemented as shown in Figure 2.17. As described in Chapter 2, the filter requires 6 integrators and 6 comb sections. The integrators operate at sampling frequency of $F_s$, since they are placed
before the downsampler. However, all the comb sections are placed after the
downsampler so that they operate at reduced frequency. In order to avoid the overflow in
the integrators the word-length is set to $6 \times \log_2 64 + 1 = 37$. The comb sections also have a
word-length of 37 since a propagation path has to be provided through the comb sections.
Table 3.3 shows the estimated power estimation of the second order sharpened CIC filter.
These values represent only the approximate estimation of the power consumption of the
sharpened CIC filter using Equation 3.2. Actual power consumption values can be
measured through circuit level simulations.

Table 3.2 Power estimation of third order non-recursive CIC filter with decimation of 64
for Case 64_1_1

<table>
<thead>
<tr>
<th>Power Calculation</th>
<th>Section-1</th>
<th>Section-2</th>
<th>Section-3</th>
<th>Section-4</th>
<th>Section-5</th>
<th>Section-6</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_s$</td>
<td>256×</td>
<td>128×</td>
<td>64×</td>
<td>32×</td>
<td>16×</td>
<td>8×</td>
</tr>
<tr>
<td>Word-length</td>
<td>4</td>
<td>7</td>
<td>10</td>
<td>13</td>
<td>16</td>
<td>19</td>
</tr>
<tr>
<td>Order</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Power Metric</td>
<td>3072</td>
<td>2688</td>
<td>1920</td>
<td>1248</td>
<td>768</td>
<td>456</td>
</tr>
</tbody>
</table>

Total Power Metric: 10152

Table 3.3 Power estimation of second order sharpened CIC filter with decimation of 64
for Case 1_1_64

<table>
<thead>
<tr>
<th>Power Calculation</th>
<th>Type</th>
<th>Number</th>
<th>Word-length</th>
<th>Rate</th>
<th>Power-metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrator</td>
<td>6</td>
<td>37</td>
<td></td>
<td>256×</td>
<td>56832</td>
</tr>
<tr>
<td>Differentiator</td>
<td>6</td>
<td>37</td>
<td></td>
<td>4×</td>
<td>888</td>
</tr>
</tbody>
</table>

Total Power Metric: 57720

Figure 3.3 Block diagram of 3rd order non-recursive CIC filter (decimation of 64)
Figure 3.4 a) Frequency response of second order sharpened CIC filter  
b) Frequency response including equiripple FIR filter

Figure 3.5 Block diagram of decimation filter

Figure 3.6 Block diagram of cascade of non-recursive and sharpened CIC filters

Although the sharpened CIC filter has negligible passband droop the total power metric of the filter is higher than the conventional CIC filter. This is due to the large word-length integrators operating at high sampling rate (this is based on Equation 3.2).
The cascade of a non-recursive CIC filter and a sharpened CIC filter with decimation of 64 addresses this issue. The following section deliberates the proposed architecture.

3.4 Case 8_8_1: Decimation of 64 using cascade of CIC and sharpened CIC filters

For Case 8_8_1, the decimation of 64 is implemented with a non-recursive CIC filter that decimates by 8, followed by a sharpened CIC filter that decimates by 8. In this case, the sharpened CIC filter can operate at a reduced sampling rate since the sampling rate is decimated by the non-recursive CIC filter. Among the possible combinations, splitting the decimation of 64 into 8 and 8 achieves the desired SNR with the lowest power metric as shown in Table 3.6. Therefore, the first stage CIC filter and the second stage sharpened CIC filter each provides a decimation of 8. Figure 3.5 shows the block diagram of decimation filter where N1 and N2 are equal to 8. Here, y1(m1) is output of the first stage CIC filter and y(m) is output of the second stage SCIC filter. y(m) is obtained by performing decimation of 64 on the input x(n). Figure 3.6 shows the block diagram of cascade of non-recursive and sharpened CIC filter. Figure 3.6 applies to all the realizations with cascade of non-recursive and sharpened CIC filter with different choices of \( H_1(z) \) and \( H_2(z) \). The z transform representation of cascade of non-recursive and sharpened CIC filter [7] is given by

\[
H_{cab}(z) = \left[H_2(z)\right]\left[H_{SCIC}(z)\right]
\]  

where \( H_2(z) \) and \( H_{SCIC}(z) \) are the transfer functions of the non-recursive CIC and sharpened CIC filters respectively. Furthermore, the sharpened CIC filter can be expressed as

\[
H_{SCIC}(z) = \left[H_1(z^{N1})\right]^2 \left\{3z^{-N1(N2-1)K/2} - 2\left[H_1(z^{N1})\right]\right\}
\]
where \( H_1(z) \) is the CIC filter and decimation rates \( N1 \) and \( N2 \) are all set to 8 in this case and \( K \) is the order of the filter. The design specifications are met using a second order non-recursive CIC filter and second order sharpened CIC filters. The frequency response of the overall structure is depicted in Figure 3.8.a. The frequency response including the equiripple FIR filter is shown in Figure 3.8.b. It exhibits a negligible passband droop of 0.0689dB, stopband attenuation of 42.9dB, and achieves SNR of 100.18dB. Implementation of the second order non-recursive CIC filter and the sharpened CIC filter is described in the following sub sections.

3.4.1 Implementation of the second order CIC filter

Figure 3.7 depicts the non-recursive implementation of second order \([H_2(z)]\) which consist of three sections with each sections having 2 copies of non-recursive blocks. Since the order of each section is 2, the word-length increases through each section by 2 bits. The first section operates at the highest sampling rate \( F_s \) whereas the last section operates at a frequency of \( F_s/4 \). Frequency of the output sequence coming out of this filter is \( F_s/8 \) as the filter does a decimation of 8. Table 3.4 shows the estimated power metric of the second order non-recursive CIC filter.

3.4.2 Implementation of the second order sharpened CIC filter

The sharpening technique applied to \([H_1(z)]\) is implemented using the architecture shown in Figure 2.17. All integrators are placed before the downsampler and all differentiators are placed after the downsampler. Hence, the differentiators operate at a reduced sampling frequency. As the order of filter is 2, cascade of 6 integrators are needed in this implementation as seen in Figure 2.17.
To account for the overflow issue in the integrators, the bit-width of all the integrators and differentiators are set to 25 ($K \log_2 N + B_m = 6 \log_2 8 + 7 = 25$). The integrators in the sharpening section operate at reduced sampling rate of $32 \times (F_s/8)$, since the first-stage [$H_2(z)$] performs a decimation of 8. As the differentiators are placed after the downsampler, they operate at sampling rate of $4 \times (F_s/64)$. Thus, the estimated total power metric for the sharpening section is 5400, which leads to a total power metric for both CIC and sharpened CIC filters of 9112.

![Frequency response of cascade of non-recursive and sharpened CIC filters](image)

![Frequency response including equiripple FIR filter (zoomed)](image)

Figure 3.7 a) Frequency response of cascade of non-recursive and sharpened CIC filters

b) Frequency response including equiripple FIR filter

![Non-recursive implementation of $H_2(z)$](image)

Figure 3.8 Non-recursive implementation of $H_2(z)$
Table 3.4 Power estimation of $H_2(z)$

<table>
<thead>
<tr>
<th>$F_z$</th>
<th>Section-1</th>
<th>Section-2</th>
<th>Section-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word-length</td>
<td>256×</td>
<td>128×</td>
<td>64×</td>
</tr>
<tr>
<td>Order</td>
<td>3</td>
<td>5</td>
<td>7</td>
</tr>
<tr>
<td>Power Metric</td>
<td>1536</td>
<td>1280</td>
<td>896</td>
</tr>
<tr>
<td>Total Power Metric</td>
<td></td>
<td></td>
<td>3712</td>
</tr>
</tbody>
</table>

Table 3.5 Power estimation of sharpened $H_1(z)$

<table>
<thead>
<tr>
<th>Type</th>
<th>Number</th>
<th>Word-length</th>
<th>Rate</th>
<th>Power-metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrator</td>
<td>6</td>
<td>25</td>
<td>32×</td>
<td>4800</td>
</tr>
<tr>
<td>Differentiator</td>
<td>6</td>
<td>25</td>
<td>4×</td>
<td>600</td>
</tr>
<tr>
<td>Total Power Metric</td>
<td></td>
<td></td>
<td></td>
<td>5400</td>
</tr>
</tbody>
</table>

3.5 Decimation of 4 using equiripple FIR filter

The remaining decimation of four is performed by an equiripple FIR filter. The filter is designed such that it has negligible passband droop and sufficiently attenuates the out of band quantization noise. The filter should have a very sharp transition band so that noise inside the main lobe of CIC filter is attenuated. The passband and stopband frequency of the filter is set to 20 KHz and 26 KHz respectively. In this case a stopband attenuation of 23dB and a passband ripple of 0.06dB are required to achieve SNR of 98dB. A 40-tap equiripple FIR filter is able to meet the desired specifications. The frequency response of the filter is depicted in Figure 3.9. The equiripple FIR filter is implemented in poly-phase transpose direct form. The poly-phase structure allows the filter to operate at rate lower than the input sampling rate, which leads to reduced power consumption.
Table 3.6 Power comparison of all the combination of decimation factors to implement decimation of 64

<table>
<thead>
<tr>
<th>Case</th>
<th>Order</th>
<th>N1</th>
<th>N2</th>
<th>N3</th>
<th>Power Metric</th>
<th>Passband droop (dB)</th>
<th>Stopband attenuation (dB)</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case-1_1_64</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>64</td>
<td>57720</td>
<td>-0.064</td>
<td>-43.76</td>
<td>100.24</td>
</tr>
<tr>
<td>Case-1_2_32</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>32</td>
<td>45432</td>
<td>-0.0639</td>
<td>-43.67</td>
<td>100.20</td>
</tr>
<tr>
<td>Case-1_4_16</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>16</td>
<td>40056</td>
<td>-0.0636</td>
<td>-43.36</td>
<td>100.20</td>
</tr>
<tr>
<td>Case-1_8_8</td>
<td>2</td>
<td>1</td>
<td>8</td>
<td>8</td>
<td>40824</td>
<td>-0.0622</td>
<td>-42.21</td>
<td>100.09</td>
</tr>
<tr>
<td>Case-1_16_4</td>
<td>2</td>
<td>1</td>
<td>16</td>
<td>4</td>
<td>45240</td>
<td>-0.0569</td>
<td>-39.28</td>
<td>99.89</td>
</tr>
<tr>
<td>Case-1_32_2</td>
<td>2</td>
<td>1</td>
<td>32</td>
<td>2</td>
<td>51768</td>
<td>-0.041</td>
<td>-37.22</td>
<td>99.31</td>
</tr>
<tr>
<td>Case-2_2_16</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>16</td>
<td>25368</td>
<td>-0.0639</td>
<td>-43.39</td>
<td>100.20</td>
</tr>
<tr>
<td>Case-2_4_8</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>23064</td>
<td>-0.0625</td>
<td>-42.23</td>
<td>100.09</td>
</tr>
<tr>
<td>Case-2_8_4</td>
<td>2</td>
<td>2</td>
<td>8</td>
<td>4</td>
<td>23640</td>
<td>-0.0572</td>
<td>-39.22</td>
<td>99.89</td>
</tr>
<tr>
<td>Case-2_16_2</td>
<td>2</td>
<td>2</td>
<td>16</td>
<td>2</td>
<td>25944</td>
<td>-0.0416</td>
<td>-37.15</td>
<td>99.31</td>
</tr>
<tr>
<td>Case-2_32_1</td>
<td>2</td>
<td>2</td>
<td>32</td>
<td>1</td>
<td>27672</td>
<td>-0.0643</td>
<td>-43.73</td>
<td>100.21</td>
</tr>
<tr>
<td>Case-4_2_8</td>
<td>2</td>
<td>4</td>
<td>2</td>
<td>8</td>
<td>15416</td>
<td>-0.0638</td>
<td>-42.32</td>
<td>100.19</td>
</tr>
<tr>
<td>Case-4_4_4</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>14456</td>
<td>-0.0585</td>
<td>-39.18</td>
<td>99.89</td>
</tr>
<tr>
<td>Case-4_8_2</td>
<td>2</td>
<td>4</td>
<td>8</td>
<td>2</td>
<td>14840</td>
<td>-0.0426</td>
<td>-36.99</td>
<td>100.11</td>
</tr>
<tr>
<td>Case-4_16_1</td>
<td>2</td>
<td>4</td>
<td>16</td>
<td>1</td>
<td>14648</td>
<td>-0.0652</td>
<td>-43.55</td>
<td>100.20</td>
</tr>
<tr>
<td>Case-8_2_4</td>
<td>2</td>
<td>8</td>
<td>2</td>
<td>4</td>
<td>10456</td>
<td>-0.0637</td>
<td>-39.07</td>
<td>100.15</td>
</tr>
<tr>
<td>Case-8_4_2</td>
<td>2</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>10072</td>
<td>-0.0474</td>
<td>-36.39</td>
<td>99.30</td>
</tr>
<tr>
<td>Case-8_8_1</td>
<td>2</td>
<td>8</td>
<td>8</td>
<td>1</td>
<td>9112</td>
<td>-0.0689</td>
<td>-42.9</td>
<td>100.18</td>
</tr>
<tr>
<td>Case-16_2_2</td>
<td>2</td>
<td>16</td>
<td>2</td>
<td>2</td>
<td>7960</td>
<td>-0.0671</td>
<td>-33.91</td>
<td>95.12</td>
</tr>
<tr>
<td>Case-16_4_1</td>
<td>2</td>
<td>16</td>
<td>4</td>
<td>1</td>
<td>6808</td>
<td>-0.0844</td>
<td>-40.06</td>
<td>96.23</td>
</tr>
<tr>
<td>Case-32_2_1</td>
<td>2</td>
<td>32</td>
<td>2</td>
<td>1</td>
<td>5864</td>
<td>-0.1485</td>
<td>-26.91</td>
<td>90.56</td>
</tr>
<tr>
<td>Case-64_1_1</td>
<td>3</td>
<td>64</td>
<td>1</td>
<td>1</td>
<td>10152</td>
<td>-0.673</td>
<td>-39.77</td>
<td>100.12</td>
</tr>
</tbody>
</table>
Figure 3.9 Frequency response of equiripple FIR filter with a decimation of 4

3.6 Comparison of all the possible filter architectures

The other possible combination of filters shown in Table 3.1 can be analyzed in a similar fashion to figure out the power metric and the performance. Table 3.6 illustrates all the possible filter implementations with a decimation factor of 64. The power metrics presented in the table does not include the equiripple FIR filter. Results show using a second order sharpened CIC filter alone consumes the highest power amongst all the other architectures. Furthermore, a third order non-recursive CIC filter alone achieves the desired SNR but has considerable passband droop which requires an additional droop compensation filter. Case 8_8_1 provides the desired SNR with the lowest power. Using a cascade of sharpened CIC filters result in a decimation filter with similar frequency response characteristics to Case 8_8_1, but with a much higher power metric.
All the possible combinations of decimation factors except Case 8_8_1 which use a cascade of non-recursive CIC and sharpened CIC filter either have SNR less than 98dB or have power metric greater than 9112. The power consumption of Case 8_8_1 is further reduced by using poly-phase implementation which is described in the following chapter. Considering Case 4_16_1, the power metric is more compared to Case 8_8_1. The poly-phase implementation of Case 4_16_1 would still be more than the poly-phase implementation of Case 8_8_1. This is because, even after poly-phase implementation, the first stage CIC filter in Case 4_16_1 operates at frequency $F_s/4$ whereas the first stage CIC filter in Case 8_8_1 operates at lower frequency $F_s/8$. Therefore, Case 8_8_1 is chosen to be the optimum architecture among the possible filter combinations.
CHAPTER IV
DECIMATION FILTER IMPLEMENTATION

Case 8_8_1 is chosen to be the lowest power solution achieving the desired SNR amongst all other cases in Chapter III. The power consumption of the decimation filter in Case 8_8_1 can be further reduced by parallel processing. Parallel processing through poly-phase decomposition is a wise way to reduce the power consumption and achieve high speed operation. The power consumption of the filter is reduced by decreasing the operation frequency of each register. Poly-phase decomposition can be applied both to the non-recursive section \([H_2(z)]\) and the sharpened section \(H_1(z)\) of the proposed filter. Various architectures for poly-phase implementation of the non-recursive section and the sharpened section are discussed in this chapter.

4.1 Implementation of \(H_2(z)\) using poly-phase decomposition

The first stage required for the implementation of case 8_8_1 is non-recursive CIC filter \([H_2(z)]\)

where

\[
H_2(z) = \left(1 + z^{-1}\right)^2 \left(1 + z^{-2}\right)^2 \left(1 + z^{-4}\right)^2.
\]

The most straight-forward implementation uses the non-recursive filter architecture shown in Figure 3.7. The non-recursive implementation of \([H_2(z)]\) consists of three stages. Each stage has a non-recursive block \((1 + z^{-1})^2\) followed by a downsampling by
As a result, the first stage operates at input frequency $F_s$, and the second stage operates at $F_s/2$, the third stage operates at $F_s/4$.

Poly-phase decomposition reduces the operating frequencies of the non-recursive stages. The power consumed by these stages can be minimized by reducing the frequencies at which each non-recursive stage operates. In this section, the non-recursive architecture shown in Figure 3.6 is decomposed into poly-phase structures so that the downsampler can be moved before the non-recursive blocks to reduce the operating frequency. There are different ways of decomposing $[H_2(z)]$ to obtain different poly-phase implementations which are described in the following sub-sections.

4.1.1 Case-I: Cascaded decimation by 2

The basic non-recursive block $(1 + z^{-1})^2$ of each stage can be rewritten as

$$(1 + z^{-1})^2 = 1 + z^{-2} + 2z^{-1} .$$

(4.2)

Assuming $E_0(z) = 1 + z^{-1}$ and $E_1(z) = 2$, the non-recursive block $(1 + z^{-1})^2$ can be expressed as

$$(1 + z^{-1})^2 = E_0(z^2) + z^{-1}E_1(z^2)$$

(4.3)

where $E_0(z^2)$ and $E_1(z^2)$ are sub-filters. Each stage shown in Figure 3.6 can be implemented using two sub-filters.

Figure 4.1 (a) shows the poly-phase implementation of Figure 3.6 using two sub-filters. The downsampler is placed after the sum of the sub-filters. Therefore, the sub-filters operate at the input frequency $F_s$, and the output sequence $y_x(mx)$ has a rate of $F_s/2$. Figures 4.1 (b) and (c) demonstrate how the downsampler is moved toward the input section. The sub-filters in Figure 4.1 (c) now operate at frequency $F_s/2$ which is
same as the frequency of the output sequence. The other stages in Figure 3.6 are also replaced with similar poly-phase structures shown in Figure 4.1 (c). As a result, the sub-filters in stages-1, 2 and 3 operate at the reduced rates of \( F_s/2, F_s/4 \) and \( F_s/8 \), respectively. Enabling the decimation to occur at the input of each sub-filter reduces the sampling frequency by 2 for each stage leading to lower power consumption.

![Diagram](image)

Figure 4.1 Poly-phase implementation of \((1 + z^{-1})^2\) using two sub-filters
(a) Downsampler outside the loop (b) Downsampler inside the loop (c) Downsampler before the sub-filters
4.1.2 Case-II: Decomposing into two sub-filters

In the previous subsection, it is shown that each non-recursive stage is implemented using two sub-filters and a downsampling by 2. Three identical stages can be cascaded to implement the \([H_2(z)]\). Instead of cascading three stages, the three stages can be combined into a single stage and downsampling by 8. The transfer function \([H_2(z)]\) is modified through Equation (4.8b). \([H_2(z)]\) is first expressed as a summation of delay elements as

\[
[H_2(z)] = \left( \frac{1-z^{-8}}{1-z^{-1}} \right)^2 = 1 + 2z^{-1} + 3z^{-2} + 4z^{-3} + 5z^{-4} + 6z^{-5} + 7z^{-6} + 8z^{-7} + 7z^{-8} + 6z^{-9} + 5z^{-10} + 4z^{-11} + 3z^{-12} + 2z^{-13} + z^{-14}.
\]  

(4.4)

Then, the delay elements are rearranged to group them. The even-indexed coefficients are grouped as the first term and the odd-indexed coefficients form the second term as

\[
[H_2(z)] = (1 + 3z^{-2} + 5z^{-4} + 7z^{-6} + 7z^{-8} + 5z^{-10} + 3z^{-12} + z^{-14})
+ z^{-1} (2 + 4z^{-3} + 6z^{-4} + 8z^{-5} + 6z^{-6} + 4z^{-10} + 2z^{-12}).
\]  

(4.5)

The first and second terms are represented as \(E_0(z^2)\) and \(E_1(z^2)\), respectively. Therefore, \([H_2(z)]\) is expressed as a sum of two sub-filters \(E_0(z^2)\) and \(E_1(z^2)\) with a delay before the second sub-filter as

\[
[H_2(z)] = E_0(z^2) + z^{-1}E_1(z^2)
\]  

(4.6)

where

\[
E_0(z^2) = 1 + 3z^{-2} + 5z^{-4} + 7z^{-6} + 7z^{-8} + 5z^{-10} + 3z^{-12} + z^{-14}
\]  

(4.7a)

and

\[
E_1(z^2) = 2 + 4z^{-3} + 6z^{-4} + 8z^{-5} + 6z^{-6} + 4z^{-10} + 2z^{-12}.
\]  

(4.7b)
Figure 4.2 Implementing $H_2(z)$ with two sub-filters

Figure 4.2 (a) depicts the implementation of $[H_2(z)]$ using two sub-filters where the two sub-filters operate at input frequency $F_s$. However, the downsampling of 2 can be moved before the sub-filters so that the sub-filters operate at a reduced frequency $F_s/2$ while maintaining the output sampling frequency of $F_s/8$. Furthermore, when the downsampling of 2 is moved before sub-filters $E_0(z^2)$ and $E_1(z^2)$, the sub-filters can be replaced by the poly-phase components $E_0(z)$ and $E_1(z)$. The poly-phase components are expressed as

$$E_0(z) = 1 + 3z^{-1} + 5z^{-2} + 7z^{-3} + 7z^{-4} + 5z^{-5} + 3z^{-6} + z^{-7}$$ \(4.8a\)

$$E_1(z) = 2 + 4z^{-1} + 6z^{-2} + 8z^{-3} + 6z^{-4} + 4z^{-5} + 2z^{-6}$$ \(4.8b\)

where $E_0(z)$ has 8 taps and $E_1(z)$ has 7 taps.

4.1.3 Case-III: Decomposing into four sub-filters

In this case, the filter $[H_2(z)]$ is implemented using four sub-filters. Consider Equation (4.4) which expresses $[H_2(z)]$ as summation of delay elements. The delay elements are now grouped into four terms as

$$[H_2(z)] = (1 + 5z^{-4} + 7z^{-8} + 3z^{-12}) + z^{-1}(2 + 6z^{-4} + 6z^{-8} + 2z^{-12})$$
$$+ z^{-2}(3 + 7z^{-4} + 5z^{-8} + z^{-12}) + z^{-3}(4 + 8z^{-4} + 4z^{-8})$$ \(4.9\).

$[H_2(z)]$ is expressed in terms of four sub-filters $E_0(z^4)$, $E_1(z^4)$, $E_2(z^4)$ and $E_3(z^4)$ as
\[ [H_2(z)] = E_0(z^4) + z^{-1}E_1(z^4) + z^{-2}E_2(z^4) + z^{-3}E_3(z^4) \] (4.10)

where

\[ E_0(z^4) = 1 + 5z^{-4} + 7z^{-8} + 3z^{-12}, \] (4.11a)

\[ E_1(z^4) = 2 + 6z^{-4} + 6z^{-8} + 2z^{-12}, \] (4.11b)

\[ E_2(z^4) = 3 + 7z^{-4} + 5z^{-8} + z^{-12} \] (4.11c)

and

\[ E_3(z^4) = 4 + 8z^{-4} + 4z^{-8}. \] (4.11d)

An efficient implementation of Equation (4.11) is depicted in Figure 4.3 (a). The data passes through appropriate delay elements before being operated by the sub-filters except for the first sub-filter \( E_0(z^4) \) which directly operates on the input data. As a result, all the sub-filters operate at frequency \( F_s \), and the adder combines the output of each sub-filter which operates at frequency \( F_s \).

Figure 4.3 (b) depicts a more efficient implementation of \([H_2(z)]\) where the downsampling of 4 is moved before the sub-filters and adder. In this case, the sub-filters \( E_i(z^4) \) can be replaced by the poly-phase components \( E_i(z) \) where \( i = 0 \) to 3 where the sub-filters \( E_0(z), E_1(z), E_2(z) \) have 4 taps and the sub-filter \( E_3(z) \) has 3 taps, as

\[ E_0(z) = 1 + 5z^{-1} + 7z^{-2} + 3z^{-3}, \] (4.12a)

\[ E_1(z) = 2 + 6z^{-1} + 6z^{-2} + 2z^{-3}, \] (4.12b)

\[ E_2(z) = 3 + 7z^{-1} + 5z^{-2} + z^{-3} \] (4.12c)

and

\[ E_3(z) = 4 + 8z^{-1} + 4z^{-2} \] (4.12d)
The data flow through the delays is same as the previous implementation of \([H_2(z)]\) shown in Figure 4.3 (a). The data are picked up from the delays simultaneously every 4\(^{th}\) instant of time, and is then fed into sub-filters for further processing. As a result, the sub-filters and the adder operate at a reduced frequency \(F_s/4\). The output of the sub-filters are summed and down sampled by 2. Therefore, the frequency of the final output \(y_1(m1)\) is \(F_s/8\).

4.1.4 Case-IV: Decomposing into eight sub-filters

Another efficient way of implementing \([H_2(z)]\) is decomposing it into eight sub-filters and performing the decimation of 8 before the sub-filters. Applying poly-phase decomposition to Equation (4.4) leads to

\[ H_2(z) = (1+7z^{-8}) + z^{-1}(2+6z^{-8}) + z^{-2}(3+5z^{-8}) + z^{-3}(4+4z^{-8}) + z^{-4}(5+3z^{-8}) + z^{-5}(6+2z^{-8}) + z^{-6}(7+z^{-8}) + z^{-7}(8) \]  

(4.13)

Now \([H_2(z)]\) is regrouped into eight sub-filters \(E_i(z^8)\) where \(i = 0\) to 7.

\[ H_2(z) = E_0(z^8) + z^{-1}E_1(z^8) + z^{-2}E_2(z^8) + z^{-3}E_3(z^8) \]
\[ + z^{-4}E_4(z^8) + z^{-5}E_5(z^8) + z^{-6}E_6(z^8) + z^{-7}E_7(z^8) \]  

(4.14)

where

\[ E_i(z^8) = (i+1) + (7-i)z^{-8} \quad i = 0 \text{ to } 6 \]  

(4.15a)

\[ E_7(z^8) = 8 \]  

(4.15b)

All the sub-filters have only 2 taps except \(E_7(z^8)\) which has a single tap. Figure 4.4 (a) depicts the implementation of Equation (4.14). Sub-filters operate at a frequency \(F_s\). However to reduce the operating frequency of the sub-filters, the downsampling of 8 can be moved before the sub-filters as shown in Figure 4.4 (b).
The sub-filters are expressed as

\[ E_i(z) = (i+1) + (7-i)z^{-1} \quad i = 0 \text{ to } 6 \]  

\[ E_f(z) = 8 \]  

The final output sequence is obtained by picking up the data from the delays every 
\((8*n)*T_s\) instant, where \(T_s = 1/F_s\) and \(n = 0, 1, 2, 3\ldots\), and passing through the sub-filters and combining the outputs with the adder. Therefore the sub-filters and the adder can operate at the frequency of \(F_s/8\).

4.2 Architectures for the implementation of sub-filters

In the previous section various architectures which decompose \([H_2(z)]\) into a group of two, four and eight sub-filters have been proposed. The sub-filters in these architectures can either be implemented using direct or transpose direct forms. The direct and the transpose direct forms can further be modified to achieve power efficient architectures. As an example, architectures implementing \([H_2(z)]\) by decomposing it into four sub-filters are deliberated. Similar architectures are developed while implementing \([H_2(z)]\) by decomposing it into two and eight sub-filters.
Consider Figure 4.3 (b) which depicts the implementation of \([H_2(z)]\) using four sub-filters. Here

\[
E_0(z) = 1 + 5z^{-1} + 7z^{-2} + 3z^{-3} \tag{4.17a}
\]

\[
E_1(z) = 2 + 6z^{-1} + 6z^{-2} + 2z^{-3} \tag{4.17b}
\]

\[
E_2(z) = 3 + 7z^{-1} + 5z^{-2} + z^{-3} \tag{4.17c}
\]

\[
E_3(z) = 4 + 8z^{-1} + 4z^{-2} \tag{4.17d}
\]

The poly-phase components \(E_0(z), E_1(z), E_2(z)\) and \(E_3(z)\) can be used in the commutative structures of Figure 4.3 (b) to implement \([H_2(z)]\). Figure 4.5 illustrates the implementation of \([H_2(z)]\) in direct form. The architecture can be improved by using coefficient symmetry in the sub-filters. In this case, sub-filters \(E_1(z)\) and \(E_3(z)\) have symmetrical coefficients. As a result, as shown in Figure 4.6 the number of multipliers can be reduced. The sub-filters can also be implemented in transpose direct form. Figure
4.7 illustrates the implementation of $[H_2(z)]$ in transpose direct form. In this architecture, the delayed and down-sampled versions of the input signal passes through the appropriate multipliers, and then through the delay chains. The output sample $y_1(m1)$ is obtained by adding the sample from the four delay chains. The individual delay chains can be combined into a single delay chain. This minimizes the power consumption by reducing the number of delays (memory elements). Figure 4.8 depicts the implementation of $[H_2(z)]$ in transpose direct form which eliminates the delay chains. Similar architectures are developed for the implementation of $[H_2(z)]$ using two and eight sub-filters. Table 4.1 presents all the poly-phase architectures that are developed to implement $[H_2(z)]$.

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>cicmFilter</td>
<td>without poly-phase</td>
</tr>
<tr>
<td>cicmFilterpoly1</td>
<td>cascaded decimation by 2</td>
</tr>
<tr>
<td>cicmFilterpoly2_1</td>
<td>direct form using 4 sub-filters</td>
</tr>
<tr>
<td>cicmFilterpoly2_11</td>
<td>direct form using 4 sub-filters exploiting symmetry</td>
</tr>
<tr>
<td>cicmFilterpoly2_12</td>
<td>transpose direct form using 4 sub-filters</td>
</tr>
<tr>
<td>cicmFilterpoly2_13</td>
<td>transpose direct form using 4 sub-filters eliminating delay chains</td>
</tr>
<tr>
<td>cicmFilterpoly2_2</td>
<td>direct form using 2 sub-filters</td>
</tr>
<tr>
<td>cicmFilterpoly2_21</td>
<td>direct form using 2 sub-filters exploiting symmetry</td>
</tr>
<tr>
<td>cicmFilterpoly2_22</td>
<td>transpose direct form using 2 sub-filters</td>
</tr>
<tr>
<td>cicmFilterpoly2_23</td>
<td>transpose direct form using 2 sub-filters eliminating delay chains</td>
</tr>
<tr>
<td>cicmFilterpoly2</td>
<td>using 8 sub-filters</td>
</tr>
<tr>
<td>cicmFilterpoly21</td>
<td>using 8 sub-filters eliminating delay blocks</td>
</tr>
<tr>
<td>cicmFilterpoly22</td>
<td>using 8 sub-filters eliminating multipliers</td>
</tr>
</tbody>
</table>
Figure 4.5 Implementation of $H_2(z)$ in direct form
Figure 4.6 Implementation of $H_2(z)$ in direct form exploiting symmetry
4.3 Implementing sharpened $H_1(z)$ using poly-phase decomposition

The $z$-domain transfer function of sharpened $H_1(z)$ is shown

$$\text{Sharpened } H_1(z) = \left[ H_1(z^{N_1}) \right]^2 \left\{ 3z^{-N_1(N_2-1)K/2} - 2 \left[ H_1(z^{N_1}) \right] \right\}$$ (4.18)
where $N_1$ and $N_2$ are the decimation factors which are equal to 8 and $K = 2$. Figure 4.9 shows the block diagram that implements Equation (4.18). The sharpened $H_1(z)$ is generally implemented as a cascade of integrators and comb sections, as shown in Figure 4.10. Therefore large word-length integrators and comb sections are required to implement the sharpened $H_1(z)$. As discussed in section 3.4.2, word-length of 25 is required for all the integrators and comb sections. Another major drawback of this architecture is the integrators that are placed before the downsampler operate at a higher sampling frequency.

The conventional architecture has been modified to eliminate the drawbacks. The proposed architecture not only reduces the operating frequency but also eliminates the large word-length integrators.

Figure 4.8 Implementation of $H_2(z)$ in transpose direct form eliminating delay chains
There are several ways to modify the implementation of the sharpened $H_1(z)$. The first approach is to implement $[H_1(z^{N1})]^2$ in Equation (4.18) as a cascade of integrators and comb sections, while implementing $\{3z^{-N1(N2-1)} - 2[H_1(z^{N1})]\}$ using poly-phase decomposition. Figure 4.11 depicts this approach. Though the poly-phase of $[H_1(z)]$ operates at reduced frequency $F_s/N_c$, the integrators still operate at higher frequency $F_s/N1$. Therefore, this architecture also suffers from the same drawback as the conventional implementation, since large word-length integrators operating at high sampling frequency consumes more power. The second approach is to move $[H_1(z^{N1})]^2$ into the loop as shown.
Sharpened $H_1(z) = \left[ 3z^{-N_l(N_2-1)} \left[ H_1(z^{N_1}) \right]^2 - 2 \left[ H_1(z^{N_1}) \right]^3 \right]$ (4.19)

$[H_1(z^{N_1})]^2$ and $[H_1(z^{N_1})]^3$ in Equation (4.19) can be implemented in poly-phase as shown in Figure 4.12. Any poly-phase structure proposed in section 4.2 can be used to implement poly-phase $[H_1(z^{N_1})]^2$ and $[H_1(z^{N_1})]^3$. Notice that the poly-phase structures of both $[H_1(z^{N_1})]^2$ and $[H_1(z^{N_1})]^3$ operate at a reduced frequency $F_s/N_c$. Therefore, the use of poly-phase structures helped in reducing the frequency of operation of sharpened $H_1(z)$ from $F_s/8$ to $F_s/64$. The transformation from conventional implementation to poly-phase also facilitated in reducing the word-lengths of all the internal registers, though the input and output number of bits are same as before. Thus this architecture not only eliminated the integrators operating at high frequency but also exploited all the advantages of using poly-phase implementation.

Figure 4.11 Implementation of the first poly-phase approach
Figure 4.12 Implementation of the second poly-phase approach
CHAPTER V
SIMULATION AND RESULTS

This chapter describes the implementation of the proposed filter using 1μm CMOS technology with a 5V supply. The performance of the filter is verified through circuit-level simulations. And the area and power consumption are compared with several different CIC and SCIC filter architectures. Using the conventional SCIC filter followed by a lowpass filter gives good frequency characteristics, at the cost of power and area. However, the cascade of CIC filter and SCIC filter with proper exploitation of the resulting two-stage decimation structure, the power consumption can be reduced compared to the conventional SCIC filter. Results show both the non-recursive and recursive CIC filters require more power compared to the proposed architecture. The Cadence RC low power (RC-LP) engine is used to implement the decimation filters and simulated using the Incisive Unified Simulator. The poly-phase structure with a commutative component that is used to implement the CIC and the SCIC filters is discussed first. Next power optimization techniques such as clock gating and operand isolation are described. The proposed architecture implementation is then shown. In addition, the poly-phase architectures described in Chapter IV (Table 4.1) are compared in terms of area and power consumption, and the proposed architecture is compared with other implementations.
Figure 5.1 Implementing $H_2(z)$ with eight sub-filters

Figure 5.2 Poly-phase implementation of transpose direct form of CIC filter
5.1 Poly-phase structure with a commutative component

The poly-phase structure shown in Figure 4.6 is shown again in Figure 5.1. As described in section 4.1.4, the structure in Figure 5.1.b has eight sub-filters and decimation of 8 is performed before each sub-filter in each channel. The data are picked up from the delays simultaneously every 8th instant of time, and is then fed into the sub-filters for further processing. The output of the sub-filters are summed to generate the output \( y_1(m1) \) running at a decimated frequency of \( F_s/8 \). The design can further be optimized by eliminating the delays blocks that are present before each channel. The input signals to each poly-phase channel \( \{x_0(m), x_1(m), \ldots, x_7(m)\} \) are the delayed and down-sampled versions of input \( x(n) \). It is straight forward to extract the 7 sequences that go into each poly-phase channel as

\[
x_0(m) = \{x(0), x(8), x(16), \ldots\} \quad (5.1)
\]
\[
x_1(m) = \{x(-1), x(7), x(15), \ldots\} \quad (5.2)
\]
\[
\ldots
\]
\[
\ldots
\]
\[
x_7(m) = \{x(-7), x(1), x(9), \ldots\} \quad (5.3)
\]

Having known the sequences that go into each channel, the sequences are directly selected from the input signal \( x(n) \) by using a commutative structure with a rotator as illustrated in Figure 5.2. The first sample \( x(0) \) goes to \( E_0(z) \), the next sample \( x(1) \) goes to \( E_7(z) \). The rotator continues in the same manner moving to the left and the sequence repeats after the rotator reaches \( E_0(z) \). Therefore, the sub-filters operate at decimated rate of \( F_s/8 \) identical to the output.
The sub-filters can be implemented in any one of the architectures as shown in Table 4.1. For instance, the sub-filters implemented in transposed direct form using a single delay chain for all the channels is shown in Figure 5.2. The input sequence \( \{x_0(m), x_1(m), \ldots, x_7(m)\} \) passes through appropriate sub-filter coefficients first and then through the delay chain to generate the final output.

5.2 Clock-gating

In most designs, the clock signal continues to toggle at every clock cycle irrespective of the frequency of the data that is loaded into the register. The clock signal often drives a large capacitive load causing large dynamic power dissipation. Clock gating technique is used to reduce the power dissipation due to clock signals driving large capacitive loads. Clock gating concept is described using a single flip-flop. A group of
such flip flops that are enabled by the same control signal are gated to reduce unnecessary clock toggles. Figure 5.3 shows a circuit with an multiplier and a D flip-flop. The multiplier has two 8-bit inputs. The circuit is controlled by enable and clock signals. The output from the D flip-flop is an 8-bit data. Figure 5.4 illustrates how clock gating is used to improve power dissipation of the circuit. The register is shut off by the gating function during the idle period to prevent power dissipation. The power is also saved in gated-clock circuitry. As several enable muxes are replaced by a single clock gating logic, clock gating insertion also reduces the area. The RC low power design (RC-LP) engine is used to explore the design to determine whether potential power savings can be made by clock gating insertion. Clock gating is inserted to those flip-flops which can result in reduced power and area without violating the timing constraints.

5.3 Operand isolation

Although clock gating results in significant reduction of power consumption, it can only save power for sequential elements and clock circuitry. Operand isolation optimizes power in datapath blocks controlled by an enable signal. Figures 5.5 and 5.6 illustrate the insertion of operand isolation. Register Rout uses the result of the multiplier only when the enable signal is on. When enable is off, register Rout uses the result of register R1. The multiplier continues its computations although its result is not used by the register Rout. Thus, significant amount of power is wasted in these multipliers. As the output of register R1 is always used either by the multiplier or the comparator, clock gating cannot be used. Therefore the functional unit (operand) is shut down (isolated) when its results are not used. AND gates are inserted at the inputs of the multiplier inputs.
The enable logic of the multiplier is used to gate the signal transitions as shown in Figure 5.6.

Figure 5.5 Circuit without operand isolation

Figure 5.6 Circuit with operand isolation

Figure 5.7 Block diagram of decimation filter
5.4 Implementation of proposed architecture

Figure 5.7 depicts the block diagram of the proposed decimation filter. The total decimation factor is 256. The input sampling frequency is $F_s = 102.4$ MHz. The first stage is a second order CIC filter with a decimation rate of 8 and the second stage is second order SCIC filter with a decimation rate of 8. The third stage is a FIR filter with a decimation rate of 4. The FIR filter described in Section 3.5 is used in the third stage. The input to decimation filter is a bit stream obtained by applying a 200 kHz sine wave input to the second order sigma-delta modulator. This frequency corresponds to the bandwidth of the ADC. All the poly-phase architectures described in Chapter IV are compared in the following section for choosing the best architecture to implement the CIC and the SCIC filters.

5.4.1 Selection of poly-phase architecture for first stage CIC filter

The first stage CIC filter is implemented using all the poly-phase architectures described in Chapter IV (Table 4.1). Conventional implementation of cascade of integrator and comb sections described in Figure 4.12 is used for second stage SCIC filter for all the implementations to make the comparison reasonable. The input and output rates for the first stage CIC filter are 102.4 MHz and 12.8 MHz respectively.

The adders and multipliers are implemented using Carry Save Adders in a Wallace tree. [23] Wallace tree has minimum propagation delay compared to other structures. Instead of completely adding the partial products in pairs like ripple carry adders, the Wallace tree sums up all the bits of the same weights in a merged tree. Weight is the $2^i$ for $i^{th}$ bit. Full adders are used to combine 3 equally weighted bits of weight $n$ to produce 2 bits: one (carry) with weight $n+1$ and other (sum) with weight $n$. Each layer of
the tree reduces the number of vectors by a factor of 3:2. Some schemes achieve a 4:2 reduction using a different adder style that adds little delay in the implementation.

Table 5.1 Area and power comparison of poly-phase architectures

<table>
<thead>
<tr>
<th>Architecture</th>
<th>AREA (mm²)</th>
<th>POWER (mW)</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>cicmFilter</td>
<td>0.109</td>
<td>4.24</td>
<td>without poly-phase</td>
</tr>
<tr>
<td>cicmFilterpoly1 Case-I</td>
<td>0.093</td>
<td>2.28</td>
<td>cascaded decimation by 2</td>
</tr>
<tr>
<td>cicmFilterpoly2_1 Case-III</td>
<td>0.095</td>
<td>2.16</td>
<td>using 4 sub-filters in direct form</td>
</tr>
<tr>
<td>cicmFilterpoly2_11 Case-III</td>
<td>0.095</td>
<td>2.24</td>
<td>using 4 sub-filters in direct form exploiting symmetry</td>
</tr>
<tr>
<td>cicmFilterpoly2_12 Case-III</td>
<td>0.096</td>
<td>2.18</td>
<td>using 4 sub-filters in transpose direct form</td>
</tr>
<tr>
<td>cicmFilterpoly2_13 Case-III</td>
<td>0.094</td>
<td>2.15</td>
<td>using 4 sub-filters in transpose direct form eliminating delay chains</td>
</tr>
<tr>
<td>cicmFilterpoly2_2 Case-II</td>
<td>0.094</td>
<td>2.24</td>
<td>using 2 sub-filters in direct form</td>
</tr>
<tr>
<td>cicmFilterpoly2_21 Case-II</td>
<td>0.096</td>
<td>2.48</td>
<td>using 2 sub-filters in direct form exploiting symmetry</td>
</tr>
<tr>
<td>cicmFilterpoly2_22 Case-II</td>
<td>0.097</td>
<td>2.13</td>
<td>using 2 sub-filters in transpose direct form</td>
</tr>
<tr>
<td>cicmFilterpoly2_23 Case-II</td>
<td>0.093</td>
<td>2.05</td>
<td>using 2 sub-filters in transpose direct form eliminating delay chains</td>
</tr>
<tr>
<td>cicmFilterpoly2 Case-IV</td>
<td>0.096</td>
<td>2.17</td>
<td>using 8 sub-filters</td>
</tr>
<tr>
<td>cicmFilterpoly21 Case-IV</td>
<td>0.095</td>
<td>2.04</td>
<td>using 8 sub-filters eliminating delay blocks</td>
</tr>
<tr>
<td>cicmFilterpoly22 Case-IV</td>
<td>0.094</td>
<td>2.27</td>
<td>using 8 sub-filters eliminating multipliers</td>
</tr>
</tbody>
</table>
D flip-flops are used for storage elements. The clock gating and the operand isolation techniques described above are used. Table 5.1 compares the area and power of all the poly-phase implementations. It can be observed from Table 5.1 that using 8 sub-filters with a single delay chain consumed the least power amongst all the poly-phase implementations. This is obvious because more decimation performed before the sub-filters lowers the sub-filter operation frequency which leads to reduced power. Eliminating the delay blocks as much as possible also helps in reducing power. Therefore if we observe each case (Case-I, Case II, Case III and Case IV) separately, implementation of sub-filters in transpose direct form eliminating the delay chains has the least power in each case. The total power reduction depends on the effectiveness of the factors mentioned above. The area is almost the same for all the poly-phase architectures. An 8 channel poly-phase implementation with the sub-filters in transposed direct form using a single delay chain as shown in Figure 5.2 is used for the first stage CIC filter.

5.4.2 Implementation of second stage SCIC filter

The modified block diagram of second stage SCIC filter proposed in Section 4.3 is shown in Figure 5.8. Poly-phase decomposition is applied to \([H_1(z)]^2\) and \([H_1(z)]^3\). Poly-phase structure shown in Figure 5.2 is used to implement \([H_1(z)]^2\) and \([H_1(z)]^3\). The multipliers 3 and -2 are moved after \([H_1(z)]^2\) and \([H_1(z)]^3\) respectively so that the multipliers operate at reduced frequency \(F_s/64\). It also contributes in reducing the number...
of bits going into $[H_1(z)]^2$ and $[H_1(z)]^3$. Carry save adders in Wallace tree are used to implement the multipliers. Any multiplication of power of 2 is done just by shifting the bits. The input sample rate of second stage SCIC filter is 12.8 MHz, and the output rate is 1.6 MHz. Wallace tree carry save adders and D flip-flops are fast enough to meet the timing requirements. The block diagram of the proposed decimation filter which is cascade of CIC and SCIC filter is shown in Figure 5.8 where the filters $[H_2(z)]$, $[H_1(z)]^2$ and $[H_1(z)]^3$ are implemented using the poly-phase structure shown in Figure 5.2.

Table 5.2 Comparison of architectures

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Passband droop (dB)</th>
<th>Stopband attenuation (dB)</th>
<th>Power (mW)</th>
<th>Area (mm$^2$)</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recursive 3$^{rd}$-order CIC</td>
<td>0.6732</td>
<td>39.77</td>
<td>02.00</td>
<td>0.074</td>
<td>100.12</td>
</tr>
<tr>
<td>Non-recursive 3$^{rd}$-order CIC</td>
<td>0.6732</td>
<td>39.77</td>
<td>01.70</td>
<td>0.070</td>
<td>100.12</td>
</tr>
<tr>
<td>Conventional 2$^{nd}$-order SCIC</td>
<td>0.0640</td>
<td>43.76</td>
<td>12.00</td>
<td>0.131</td>
<td>100.24</td>
</tr>
<tr>
<td>Proposed</td>
<td>0.0689</td>
<td>42.90</td>
<td>01.05</td>
<td>0.116</td>
<td>100.18</td>
</tr>
</tbody>
</table>

5.5 Comparison of architectures

The conventional SCIC filter and the proposed modified architecture are implemented in a 1µm CMOS technology with a 5V supply. The input sampling frequency is $F_s = 102.4$ MHz. The simulation results are shown in Table 5.2. The power-consumption and SNR values in the table were determined by circuit-level simulation of the filters. The power and area values in the table do not include the last stage FIR filter. The filter input was a bit stream obtained by applying a 200 kHz sine-wave input to a sigma-delta modulator. This frequency corresponds to the bandwidth of the filter. FFT of
the output of the decimation filter is analyzed to measure the SNR values. SNR is measured as the difference between the signal power and the noise power (in dB).

The conventional SCIC filter followed by a low pass filter gives good frequency characteristics, but at the cost of power and area. It consumes 12 mW and occupies 0.131 mm², whereas the proposed architecture consumes only 1.05 mW and occupies 0.116 mm². The power and area savings are achieved with some compromise in filter frequency characteristics, as indicated by the slightly larger passband droop and smaller stopband attenuation shown in Table 5.2. The conventional SCIC filter and the proposed architecture are further compared with non-recursive and recursive CIC filters. The non-recursive and recursive third order CIC filters are also implemented with the same design specifications. Both the non-recursive and recursive CIC filters require more power than the proposed architecture.
CHAPTER VI
CONCLUSION AND FUTURE WORK

6.1 Summary

This thesis presents the design of a low power sharpened CIC decimation filter for Sigma-Delta ADC in a 1µm CMOS technology with a 5V supply. The cascaded filter architecture consists of the first stage CIC fitter and the second stage SCIC filter. The proposed scheme is finalized by finding the optimum decimation factor of each stage that gives minimum power consumption, whereas maintaining the passband and stopband characteristics. In addition, the power consumption of the proposed filters was further reduced by poly-phase implementation of each stage. An 8 channel poly-phase implementation with the sub-filters in transposed direct form is found to have minimum power consumption amongst all other poly-phase implementations. The proposed architecture consumes a power of 1.05 mW and occupies an area of 0.11 mm². The proposed design shows the lowest power consumption among similar filter architecture with equivalent performance.

6.2 Future work

The proposed work can be extended to decimation filters of any order and any decimation factor. Future work involves extending the proposed design to implement a programmable decimating digital filter. The proposed design has negligible passband droop for any filter order and decimation factor in contrast to conventional CIC filters which have variable passband droop. As a result, CIC filters cannot be used in
programmable decimating digital filter as they require programmable FIR filters. In contrast, only a fixed coefficient FIR filter can be used with the proposed design to extract the passband of interest and further reduce the decimation factor. This could be of a huge advantage since fixed coefficient FIR filters save a lot of power and area compared to a programmable FIR filter.
BIBLIOGRAPHY


[8] Xiong Liu, A.N. Willson, “A 1Gsample/Sec non-recursive sharpened cascaded integrator-comb filter with 70 dB alias rejection and 0.003 dB droop in 0.18-µm CMOS,” *IEEE 8th International Conference on ASIC*, pp. 867-870, October 2009.


