LOW POWER DIGITAL FILTER IMPLEMENTATION IN FPGA FOR HEARING AID APPLICATION

Jitendra Kumar Das

Department of Electronics & Communication Engineering
National Institute of Technology Rourkela

LOW POWER DIGITAL FILTER IMPLEMENTATION IN FPGA FOR HEARING AID APPLICATION

A thesis submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy
in
Electronics & Communication Engineering

By

Jitendra Kumar Das
Roll No: 50609001

Under the supervision of

Dr. Kamala Kanta Mahapatra
Professor

Department of Electronics & Communication Engineering
National Institute of Technology Rourkela

July, 2009

Dedicated

to

……. my Parents ……

Department of Electronics & Communication Engineering
NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA
ORISSA, INDIA – 769 008

CERTIFICATE

This is to certify that the thesis titled "***Low Power Digital Filter Implementation in FPGA for Hearing Aid Application***", submitted to the National Institute of Technology, Rourkela by **Jitendra Kumar Das**, Roll No. **50609001** for the award of the degree of ***Doctor of Philosophy*** in Electronics & Communication Engineering, is a bona fide record of research work carried out by him under my supervision and guidance.

The candidate has fulfilled all the prescribed requirements.

The thesis, which is based on candidate's own work, has not been submitted elsewhere for the award of a degree.

In my opinion, the thesis is of the standard required for the award of Doctor of Philosophy in Electronics & Communication Engineering.

To the best of my knowledge, he bears a good moral character and decent behaviour.

Prof. K. K. Mahapatra
Professor
Department of Electronics & Communication Engineering
NATIONAL INSTITUTE OF TECHNOLOGY
Rourkela-769 008 (INDIA)
Email: kkm@nitrkl.ac.in

ACKNOWLEDGEMENT

I would like to take this opportunity to extend my deepest gratitude to my teacher and supervisor, Prof. K. K. Mahapatra, for his continuous encouragement and active guidance. I am indebted to him for the valuable time he has spared for me during this work. He is always there to meet and talk about my ideas, to proofread and mark up my research papers and chapters, and to ask me good questions to help me think through my problems.

I am very much thankful to Prof. S. K. Patra, HOD, ECE Department for his continuous encouragement. Also, I am indebted to him who provided me all official and laboratory facilities.

I am grateful to Dr. S. Meher and Prof. T.K. Dan for his valuable suggestions and comments during this research period.

My sincere thanks go to Prof. G.S. Rath, Prof. S. K. Jena, Prof. A. K. Panda and Prof. B.D. Sahoo whose valuable suggestions helped me a lot in completing this thesis.

In addition, let me thank all my friends Saroj, Pankaj, Sushant, Manas, Debi Mohanty, Nilamani Bhoi, Sudeendra Kumar, Ayas Kanta and Peter for their great support and encouragement during the research period. Also, I am thankful to all the non-teaching staffs of ECE Department for their kind cooperation.

During the course of this work, I am supported by a project VLSI-SMDP sponsored by DIT, Govt. of India. I am really thankful to them.

Last but not the least, I take this opportunity to express my regards and obligation to my father-in-laws and family members for encouraging me in all expects. I would also like to thankful my wife *Dugulu* and son *Omm* for their unconditional support and encouragement in carrying my Ph. D work.

Jitendra Kumar Das

BIO-DATA OF THE CANDIDATE

Name of the Candidate	:	Jitendra Kumar Das
Father's Name	:	Gopinath Das
Permanent Address	:	AT/PO- Nuapatna(T) PS- Tigiria, Cuttack Orissa 754035
Email ID	:	jkdas12@gmail.com

ACADEMIC QUALIFICATION

- Continuing **Ph. D.** in Electronics & Communication Engineering, National Institute of Technology Rourkela, Orissa (INDIA). Year of degree awarded or expected -2009-10.

- **M. Tech.** in Electrical Engineering (Electronics System and Communication), National Institute of Technology Rourkela, Orissa (INDIA).

- **B. E.** (Electronics and Telecommunication), Utkal University, Orissa, (INDIA).

PUBLICATION

- Published/ 04 papers in National and International Conferences which are listed in page 164 as "Contribution by the candidate".

Abstract

Digital filters suitable for hearing aid application on low power perspective have been developed and implemented in FPGA in this dissertation.

Hearing aids are primarily meant for improving hearing and speech comprehensions. Digital hearing aids score over their analog counterparts. This happens as digital hearing aids provide flexible gain besides facilitating feedback reduction and noise elimination. Recent advances in DSP and Microelectronics have led to the development of superior digital hearing aids. Many researchers have investigated several algorithms suitable for hearing aid application that demands low noise, feedback cancellation, echo cancellation, etc., however the toughest challenge is the implementation. Furthermore, the additional constraints are power and area. The device must consume as minimum power as possible to support extended battery life and should be as small as possible for increased portability. In this thesis we have made an attempt to investigate possible digital filter algorithms those are hardware configurable on low power view point.

Suitability of decimation filter for hearing aid application is investigated. In this dissertation decimation filter is implemented using 'Distributed Arithmetic' approach. While designing this filter, it is observed that, comb-half band FIR-FIR filter design uses less hardware compared to the comb-FIR-FIR filter design. The power consumption is also less in case of comb-half band FIR-FIR filter design compared to the comb-FIR-FIR filter. This filter is implemented in Virtex-II pro board from Xilinx and the resource estimator from the system generator is used to estimate the resources.

However 'Distributed Arithmetic' is highly serial in nature and its latency is high; power consumption found is not very low in this type of filter implementation. So we have proceeded for 'Adaptive Hearing Aid' using Booth-Wallace tree multiplier. This algorithm is also implemented in FPGA and power calculation of the whole system is done using Xilinx Xpower analyser. It is observed that power consumed by the hearing aid with Booth-Wallace tree multiplier is less than the hearing aid using Booth multiplier (about 25%). So we can conclude that the hearing aid using Booth-Wallace tree multiplier consumes less power comparatively.

The above two approached are purely algorithmic approach. Next we proceed to combine circuit level VLSI design and with algorithmic approach for further possi-

ble reduction in power.

A MAC based FDF-FIR filter (algorithm) that uses dual edge triggered latch (DET) (circuit) is used for hearing aid device. It is observed that DET based MAC FIR filter consumes less power than the traditional (single edge triggered, SET) one (about 41%). The proposed low power latch provides a power saving upto 65% in the FIR filter. This technique consumes less power compared to previous approaches that uses low power technique only at algorithmic abstraction level.

The DET based MAC FIR filter is tested for real-time validation and it is observed that it works perfectly for various signals (speech, music, voice with music). The gain of the filter is tested and is found to be 27 dB (maximum) that matches with most of the hearing aid (manufacturer's) specifications. Hence it can be concluded that FDF FIR digital filter in conjunction with low power latch is a strong candidate for hearing aid application.

List of Abbreviations

ADC	*Analog to Digital convertor*
ASIC	*Application Specific Integrated Circuits*
CIC	*Completely- in Canal*
CLA	*Carry Look Ahead Adder*
CPA	*Carry Propagate adder*
CSA	*Carry Save Adder*
CSAT	*Carry Save Adder Tree*
DAC	*Digital to Analog Converter*
DET	*Dual-edge Triggered*
DETSE	*Dual-edge Triggered Storage Elements*
DSP	*Digital Signal Processing*
FDF	*Folded Direct Form*
FF	*Flip-flop*
FFs	*Flip-flops*
FIR	*Finite Impulse Response*
GPP	*General Purpose processor*
IO	*Input Output*
LMS	*Least Mean Square*
LSB	*Least Significant Bit*
LUT	*Look up Table*
MAC	*Multiply and accumulate*
MSB	*Most Significant Bit*
MSI	*Medium Scale Integration*
MUX	*Multiplexer*
PCM	*Pulse Code Modulation*
RLS	*Recursive Least Square*
RT*L*	*Register Transfer Level*
SET	*Single-edge Triggered*
SETSE	*Single Edge Triggered Storage Elements*
VHDL	*Very High Speed Integrated Circuit Description Language*
VLSI	*Very Large Scale Integration*

List of Figures

Contents

Page no.

Chapter – I

Introduction

1.1 Hearing Aids

A hearing aid is a small electronic device that one wears in or behind his/her ear. It makes sounds louder so that a person with hearing loss can listen, communicate, and participate better in daily activities. A hearing aid can help people hear more in both quiet and noisy situations. A hearing aid has three basic parts: a microphone, amplifier, and speaker. The hearing aid receives sound through a microphone, which converts the sound waves to electrical signals and sends those to an amplifier. The amplifier increases the power of the signals and then sends to the ear through a speaker.

Hearing aids are primarily useful in improving the hearing and speech comprehension of people who have hearing loss that results from damage to the small sensory cells in the inner ear, called hair cells. The damage can occur as a result of disease, aging, or injury from noise or certain medicines.

A hearing aid magnifies sound vibrations entering the ear. Surviving hair cells detect the larger vibrations and convert them into neural signals that are passed along to the brain. The greater the damage to a person's hair cells, the more severe is the hearing loss, and the greater the hearing aid amplification needed to be larger. However, there are practical limits to the amount of amplification a hearing aid can provide. In addition, if the inner ear is too damaged, even large vibrations will not be converted into neural signals. In this situation, a hearing aid would be ineffective [1].

There are three basic styles of hearing aids. The styles differ by size, their placement on or inside the ear, these are

- **Behind-the-ear** (BTE) hearing aids consist of a hard plastic case worn behind the ear and connected to a plastic earmold that fits inside the outer ear.
- **In-the-ear** (ITE) hearing aids fit completely inside the outer ear and are used for mild to severe hearing loss.
- **Canal** aids fit into the ear canal and are available in two styles. The in-the-canal (ITC) hearing aid is made to fit the size and shape of a person's ear canal. A completely-in-canal (CIC) hearing aid is nearly hidden in the ear canal. Both types are used for mild to moderately severe hearing loss.

Awareness on hearing aids and its design [2] has made an important topic of research. Acoustic noise, acoustic feedback problems in the hearing aids have become more noticeable. In daily life undesirable effect of noise, acoustic feedback in hearing aid is very annoying. The use of modern technology, light weight, low power device has made an impact in the development of hearing aids. In this dissertation we have made an attempt to design a class of low power digital filter suitable for hearing aid application.

1.2 Motivation

Approximately 10% of the population suffers from some hearing loss, however only a small percentage of these categories actually use a hearing aid device. There are several factors affecting market penetration like

a. Stigma associated with wearing the device.
b. Customer dissatisfaction with the devices not meeting their performances.
c. Cost associated with devices.

Rapid advancements in DSP and Microelectronics have facilitated the growth and use of hearing aids and its related research. DSP techniques have certain advantages like

a. Less sensitive to component parameters and environmental parameters.
b. It allows changing the processor characteristics during processing such as implementation of adaptive filters.
c. DSP can be applied to low frequency signals (Hearing Aid operates physically from 300 Hz to 4 kHz).

With these advantages, DSP provides an efficient way for managing signal processing in our daily life. Recent development of commercial hearing aid devices and DSP technique has allowed the developer to accommodate advanced signal processing techniques in hearing aid devices. This results in accurate reproduction of the sound with minimum distortion. Almost all major hearing aid manufacturers have digital hearing aid products in the markets, but only 20% of those devices are acquired by different users. This is due to large size and high power consumption. To mitigate the above

mentioned problems, it is required to find suitable low power algorithms that should additionally meet area constraints in VLSI design.

Hence the most serious issue of hearing aid device is that it should consume low power. To investigate this issue, we have investigated three different types of filter in designing low power filters. All filter designs are investigated and a structure has been proposed for low power implementation. Since hardware realization is important, all the structures in this thesis have been implemented in FPGA for testing its operation and power estimation is being carried out in each case. To bring out further insight real-time signal are applied to one of the structure for further validation. Therefore, basically we need to investigate

a. Algorithm
b. Low power VLSI.

1.3 Signal Processing

Signal processing is the analysis, interpretation, and manipulation of signals. Signals are electrical representations of time-varying or spatial-varying physical quantities, either analog or digital, and may come from various sources. Signals of interest include speech, sound, images, radar signals, and many others. Processing of such signals includes filtering, storage and reconstruction, separation of information from noise (for example, aircraft identification by radar), compression (for example, image compression), and feature extraction (for example, speech-to-text conversion). Following are the two subfields of signal processing [3].

a. Analog Signal Processing
b. Digital Signal Processing

Digital signal processing is an area of science and engineering that has developed rapidly due to the significant advances in digital computer technology and integrated circuit fabrication. The rapid developments in integrated-circuit technology, starting from MSI to VLSI of electronic circuit has spurred the development of powerful, smaller, faster, and cheaper digital computers and special-purpose digital hardware. These inexpensive and relatively fast digital circuits have made it possible to construct highly sophisticated digital systems capable of performing complex digital signal processing functions and tasks, which are usually too difficult and/or too

expensive to be performed by analog circuitry or analog signal processing systems. Hence many of the signal processing tasks that were conventionally performed by analog means are realized today by less expensive and often more reliable digital hardware.

Not only do digital circuits yield cheaper and more reliable systems for signal processing they have other advantages as well. In particular, digital processing hardware allows programmable operations. Through software, one can more easily modify the signal processing functions to be performed by the hardware without modifying the hardware. Thus digital hardware and associated software provide a greater degree of flexibility in system design. Also, there is often a higher order of precision achievable with digital hardware and software compared with analog circuits and analog signal processing systems.

For all these reasons, there has been an explosive growth in digital signal processing theory and applications over the past three decades. DSP is a key enabling technology for many applications in fields such as telecommunications, consumer electronics, hearing aid devices, disk drives, and navigation. DSP functions can be implemented using a range of implementation approaches: Application Specific Integrated Circuit (ASIC), Fixed Point ASIC (FASIC), general-purpose processors, and programmable digital signal processors are all commonly used.

In this connection, a digital hearing aid uses variety of advanced DSP algorithms such as noise reduction or echo cancellation. For this customized DSP processors or generic DSP cores can be used. Products available in the market today are all based on customized DSP cores as power dissipation is an important issue in it. Although, analog signal processors are available, it is not generally used, as we discuss here.

1.3.1 Advantages of Digital over Analog Signal Processing

Digital signal processing techniques have numerous advantages. Digital circuits are not dependent on precise values of digital signals for their operation. Digital circuits are less sensitive to changes in component values. They are also less sensitive to variations in temperature, ageing and other external parameters. Digital processing of a signal facilitates the sharing of a single processor among a number of signals by time-sharing. This reduces the processing cost. In addition multirate processing is possible only in digital domain. Storage of digital data is very easy. Digital process-

ing is much more suited for processing very low frequency signals [3-4].

1.3.2 Basic Elements of a Digital Signal Processing Systems

Most of the signals encountered in science and engineering are analog in nature. That is, the signals are functions of a continuous variable, such as time or space, and usually take on values in a continuous range. Such signals may be processed directly by appropriate analog systems (such as filters or frequency analyzers or frequency multipliers) for the purpose of changing their characteristics or extracting some desired information. In such a case we say that the signal has been processed directly in its analog form, as shown in figure 1.1.

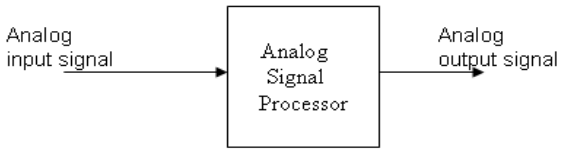


Figure 1. 1: Analog signal processing

Both the input signal and the output signal are in analog form. Digital signal processing provides an alternative method for processing the analog signal as shown in figure 1.2. To perform the processing digitally, there is a need for an interface between the analog signal from outside world and the digital processor. This interface is called an analog to digital converter. The output of the A/D converter is a digital signal that is appropriate as an input to the digital processor [3-4].

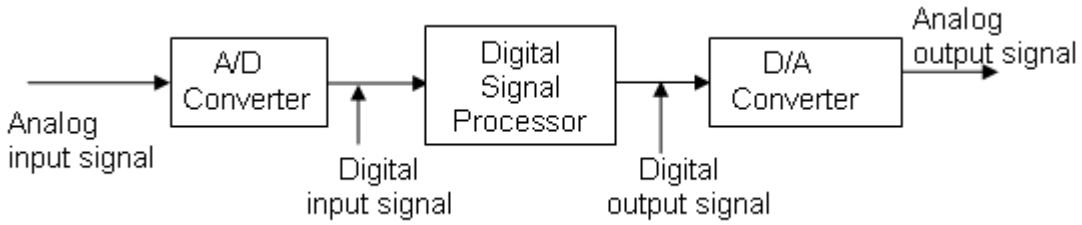


Figure 1. 2: Block diagram of a digital signal processing system.

The digital signal processor may be a large programmable digital computer or a small microprocessor programmed to perform the desired operations on the input signal. It may also be a hardwired digital processor configured to perform a specified set of operations on the input signal. Programmable machines provide the flexibility to change the signal processing operations through a change in the software, whereas hardwired machines are difficult to reconfigure.

Since in this dissertation various DSP algorithms would be handled, it is important to have a clear understanding of the DSP processor.

1.4 DSP Processor

A DSP processor is designed to support fast execution of the repetitive, numerically intensive computations characteristic of digital signal processing algorithms. The most often cited of these features is the ability to perform a multiply-accumulate operation (often called a "MAC") in a single instruction cycle. A single-cycle MAC operation is extremely useful in algorithms that involve computing a vector dot-product, such as digital filters. Such algorithms are very common in DSP applications. To achieve a single-cycle MAC, all DSP processors include a multiplier and accumulator as central elements of their data-paths [3-4].

A second feature shared by DSP processors is the ability to complete several accesses to memory in a single instruction cycle. This allows the processor to fetch an instruction while simultaneously fetching operands for the instruction, and/or storing the result of the previous instruction to memory. Typically, multiple memory accesses in a single cycle are possible only under restricted circumstances.

To allow numeric processing to proceed quickly, DSP processors incorporate one or more dedicated address generation units. The address generation units operate in parallel with the execution of arithmetic instructions, forming the addresses required for data memory accesses. The address generation units typically support addressing modes tailored to DSP applications.

Because many DSP algorithms involve performing repetitive computations, most DSPs provide hardware support for efficient looping. Often, a special loop or repeat instruction is provided which allows the programmer to implement for next loop without expanding any instruction cycles for updating and testing the loop counter and branching to the top of the loop. Finally, to allow low-cost, high-performance input and output, many DSPs incorporate one more serial or parallel I/O interfaces, and specialized I/O handling mechanisms such as low- overhead interrupts or DMA.

1.4.1 Fixed Versus Floating Point number

DSP chip word size determines resolution and dynamic range. In the fixed point processors, a linear relationship exists between word size and dynamic range. The fixed

point DSPs have either 16 or 24 bit data bus. There are four common ways to represent 2^{16} = 65,536 possible bit patterns for a number. In unsigned integer, the stored number can take on any integer value from 0 to 65,535. Similarly, signed integer uses two's complement to make the range include negative numbers, from -32,768 to 32,767. With unsigned fraction notation, the 65,536 levels are spread uniformly between 0 and 1. Lastly, the signed fraction format allows negative numbers, equally spaced between -1 and 1.

The floating point chip performs integer or real arithmetic. Normally, floating point DSP formats are 32 data bits wide and in which 24 bits form the mantissa and 8 bits make up the exponent. This results in many more bit patterns than for fixed point, 2^{32} = 4,294,967,296 to be exact. A key feature of floating point notation is that the represented numbers are not uniformly spaced. All floating point DSPs can also handle fixed point numbers, a necessity to implement counters, loops, and signals coming from the ADC and going to the DAC. However, this doesn't mean that fixed point math will be carried out as quickly as the floating point operations; it depends on the internal architecture.

Fixed point arithmetic is much faster than floating point in general purpose computers. However, with DSPs the speed is about the same, a result of the hardware being highly optimized for math operations. The internal architecture of a floating point DSP is more complicated than for a fixed point device. All the registers and data buses must be 32 bits wide instead of only 16; the multiplier and ALU must be able to quickly perform floating point arithmetic, the instruction set must be larger (so that they can handle both floating and fixed point numbers), and so on. Floating point (32 bit) has better precision and a higher dynamic range than fixed point (16 bit). In addition, floating point programs often have a shorter development cycle, since the programmer doesn't generally need to worry about issues such as overflow, underflow, and round-off error. On the other hand, fixed point DSPs have traditionally been cheaper than floating point devices.

1.4.2 Architecture of Digital Signal Processor

Although fundamentally related, DSP processors are significantly different from general purpose processors (GPPs). To understand why, we need to know what is involved in signal processing. Some of the most common functions performed in the digital domain are signal filtering, convolution and fast Fourier transform. In mathe-

matical terms, these functions perform a series of dot products. This brings us to the most popular operation in DSP: the multiply and accumulate (MAC).

The first major architectural modification that distinguished DSP processors from the early GPPs was the addition of specialized hardware that enabled single-cycle multiplication. DSP architects also added accumulator registers to hold the summation of several multiplication products. Accumulator registers are typically wider than other registers, often providing extra bits, called guard bits, to avoid overflow. Typical DSP algorithms require more memory bandwidth than the Von Neumann architecture used in GPPs. Thus, most DSP processors use some forms of Harvard architecture which has two separate memory spaces, typically partitioned as program and data memories.

Although, this may seem that DSP applications must pay careful attention to numeric accuracy - which is much easier to do with a floating-point data path, fixed-point machines tend to be cheaper (and faster) than comparable floating-point machines. To maintain accuracy without the complexity of a floating-point data path, DSP processors usually include a good support for saturation arithmetic, rounding, and shifting.

Another distinction of DSP processors is specialized addressing modes that are useful for common signal-processing operations and algorithms. The generic architecture of a DSP processor is shown in figure 1.3. The architecture has two separate memory spaces (program and data) which can be accessed simultaneously. This is similar to the Harvard architecture employed in most of the programmable DSPs. The arithmetic unit performs fixed point computation on numbers represented in “2’s” complement form. It consists of a dedicated hardware multiplier and an adder/subtractor connected to the accumulator so as to be able to efficiently execute the multiply-accumulate (MAC) operation.

In our dissertation we will use different digital filters for hearing aid application; we discuss that in the next section.

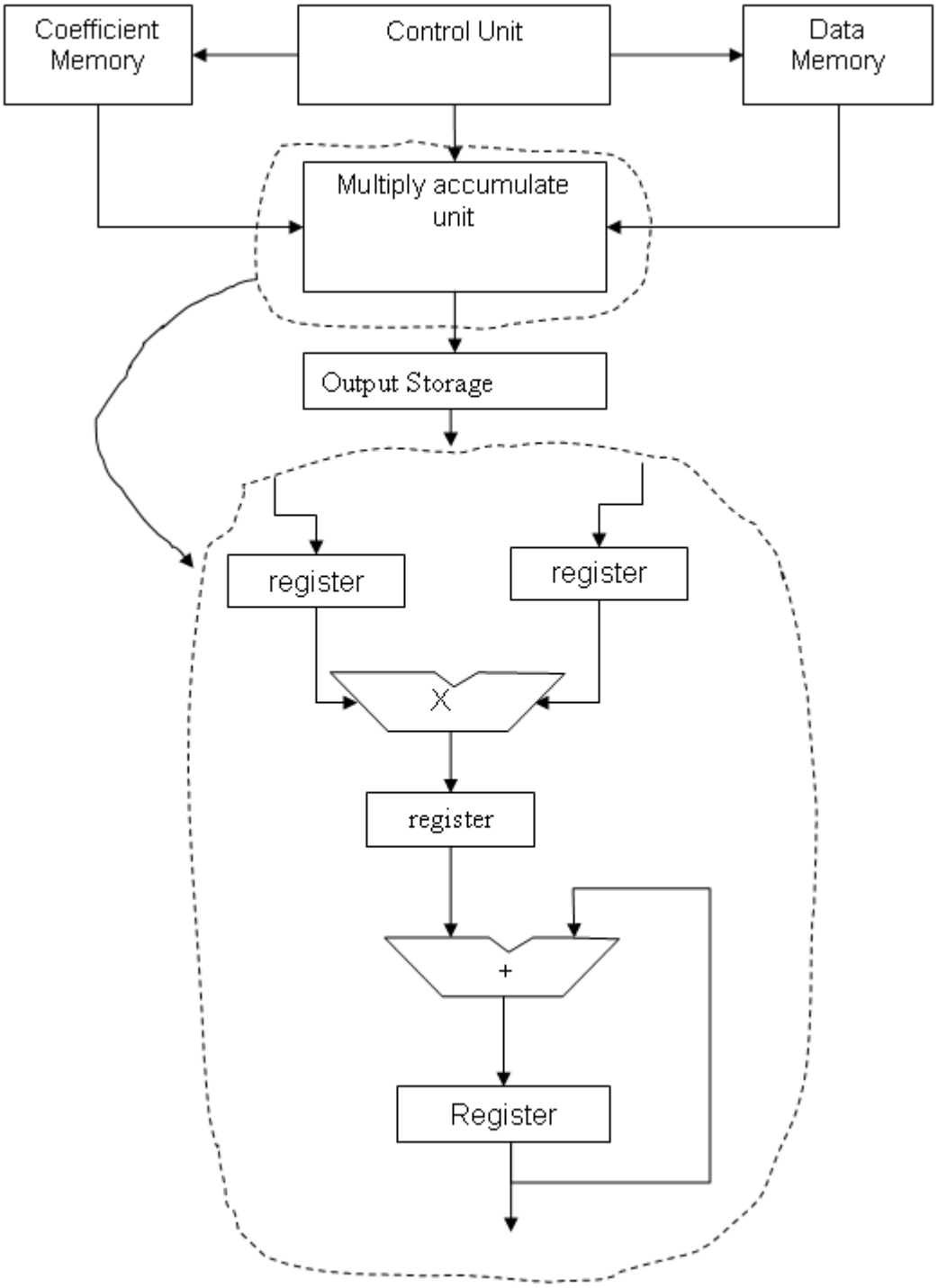


Figure 1. 3: Generic DSP processor architecture

1.5 Digital Filters

1.5.1 Analog and Digital filters

In signal processing, the function of a filter is to remove unwanted parts of the signal, such as random noise, or to extract useful parts of the signal, such as the components lying within a certain frequency range. The following block diagram in figure 1.4 illustrates the basic idea.

Figure 1. 4: Basic idea of a filter

There are two main kinds of filter, analog and digital. They are quite different in their physical makeup and in how they work. An analog filter uses analog electronic circuits made up from components such as resistors, capacitors and op-amps to produce the required filtering effect. Such filter circuits are widely used in such applications as noise reduction, video signal enhancement, graphic equalizers in hi-fi systems, and many other areas. There are well-established standard techniques for designing an analog filter circuit for a given requirement. At all stages, the signal being filtered is an electrical voltage or current which is the direct analogue of the physical quantity (e.g. a sound or video signal or transducer output) involved.

A digital filter uses a digital processor to perform numerical calculations on sampled values of the signal. The processor may be a general-purpose computer such as a PC, or a specialized DSP (Digital Signal Processor) chip. The analog input signal must first be sampled and digitized using an ADC (analog to digital converter). The resulting binary numbers, representing successive sampled values of the input signal, are transferred to the processor, which carries out numerical calculations on them. These calculations typically involve multiplying the input values by constants and adding the products together. If necessary, the results of these calculations, which now represent sampled values of the filtered signal, are output through a DAC (digital to analog converter) to convert the signal back to analog form. In a digital filter, the signal is represented by a sequence of numbers, rather than a voltage or current. The figure 1.5 shows the basic setup of such a system.

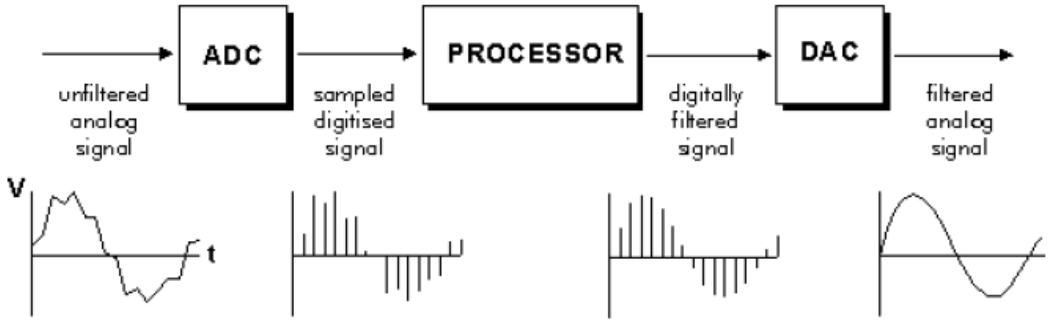


Figure 1. 5: Basic set-up of a digital filter

1.5.2 Advantages of digital filters

The following list gives some of the main advantages of digital filters over analog counterpart [3].

- A digital filter is programmable, i.e. its operation is determined by a program stored in the processor's memory. This means the digital filter can easily be changed without affecting the circuitry (hardware). An analog filter can only be changed by redesigning the filter circuit.
- Digital filters are easily ***designed, tested*** **and** ***implemented*** on a general-purpose computer or workstation.
- The characteristics of analog filter circuits (particularly those containing active components) are subject to ***drift*** and are dependent on temperature. Digital filters do not suffer from these problems, and so are extremely ***stable*** with respect both to time and temperature.
- Unlike their analog counterparts, digital filters can handle ***low frequency*** signals accurately. As the speed of DSP technology continues to increase, digital filters are being applied to high frequency signals in the RF (radio frequency) domain, which in the past was the exclusive preserve of analog technology.
- Digital filters are very much more ***versatile*** in their ability to process signals in a variety of ways; this includes the ability of some types of digital filter to adapt to changes in the characteristics of the signal.
- Fast DSP processors can handle complex combinations of filters in parallel or cascade (series), making the hardware requirements relatively ***simple*** and ***compact*** in comparison with the equivalent analog circuitry.

1.6 Algorithms for Hearing Aid Design

Most of the hearing devices use modern signal processing techniques in order to reduce noise or feedback signal that results from the secondary path i.e the sound signal that propagates from the speaker to the microphone. This causes undesired howling effects. From this point of view we have carried out a series of literature review on signal processing used in hearing aid devices.

1.6.1 Feedback cancellation in hearing aids

Feedback cancellation can be performed in various ways like

a. constrained and unconstrained adaption
b. open loop and closed loop identification
c. bias and unbiased feedback cancellation

In constrained adaption, when applying constraint on the magnitude of the adaptive weight vectors, it greatly reduces the probability that adaptive filter will cancel the narrowband input signals [5]. This paper indicates that constrained adaption has better efficiency than the unconstrained adaption.

Feedback cancellation that employs two types of open loop identifications is being reported in the literature [6]. The first algorithm uses second order signal statistics to adapt the weights while the second algorithm uses higher order statistics to adapt the weights. According to the theory; higher order statistics should be able to perform better to eliminate white noise and pink noise, but it has been shown that the higher order statics fails when the SNR of input signal falls below 0 dB [6]. Finally it has been concluded that in this case second order statistics performs better than higher order statistics in speech enhancement of hearing aid devices.

Acoustic feedback in hearing aid devices reduces the maximum usable gain. In continuous adaption of feedback cancellation, a bias is found in the adaptive filter's estimate in feedback estimation. It has been reported that using a delay in the forward or cancellation path of the hearing aid device, bias can be reduced by 15 dB [7].

Use of fixed length FIR filters in feedback path with linear gain in hearing aids has been reported and can avoid instability and howling effect in everyday use [8]. Undesired effect of the acoustic feedback in hearing aids can be reduced with an internal feedback path which gives a limiting estimate of the external feedback path.

In paper [9] LMS or RLS algorithm is used and it has been shown that limiting estimate will be biased if there exists an error in the model. To mitigate this problem, a second signal is added to the output of the hearing aid to avoid the non linearity of the hearing aid. Frequency domain adaptive filtering is used for this analysis.

Psychoacoustic Approach has been applied to hearing aid problems and in this method use of a post-filter used in the forward path has been reported [10]. This paper also reports that psycho-acoustically motivated weighting rule method of adaption is good for getting natural near end speech and results in less annoying residual noise.

Some authors use non-uniform digital FIR filter bank [11] for hearing aid application. This filter is designed by taking eight non-uniform spaced subbands that uses frequency response masking technique to give a stopband attenuation of 80 dB. It is reported that, non-uniform digital filter banks achieve good matching between audiograms and magnitude response of the filter banks with a low computational cost.

Multi delay line filter applications are reported [12] in which smaller transforms are used and ensures better control over stability obtained. Feedback cancellation in hearing aids based on Filtered-X LMS has also been reported [13]. The identification is thus done in closed loop. In this case optimal estimate is biased when the identification is performed in closed loop and the input signal to the hearing aid is not white. The bias could be avoided if the spectrum of the input signal was known and the data used to update the internal feedback is pre-filtered. The effects of different choices of the design variables of the Filtered-X LMS are taken into account and the optimal estimate of the Filtered-X LMS with an individually adjusted fixed filter showed the best agreement with the desired estimate.

Previous discussion of feedback cancellation uses single channel for feedback cancellation. When a stereophonic echo cancellation is used, there exists non-unique solution [14]. The non-uniqueness problem exists due to coherence between the two incoming audio channels. One proven solution to this problem is to distort the signals with a nonlinear device. This presents a novel theory that gives an insight to the existing links between: i) Coherence and level of distortion, and ii) coherence and achievable misalignment of the stereophonic echo canceller. Furthermore, when an adaptive nonlinear device is used for adaption, a pre-specified maximum misalignment is maintained while improving the perceived quality by minimizing the introduced distortion [14].

Howling is very annoying problem to the hearing-aid users that limits the maximum usable gain of hearing-aid devices. The paper [15] , presents an effective feedback cancellation system where a time-varying all-pass filter together with a delay in the forward path is used for de-correlating the input and output signals of the hearing aid plant. The adaptive filter in the hearing aid performs continuous adaptation based on the input signal. The output signal of the hearing aid is processed by a 2nd-order APF whose frequency is varied using a low-frequency modulator. It has been reported that the time varying all pass filter can significantly reduce the weight-vector misalignment with a small delay in the forward path.

Most of the above discussion uses narrowband signals for feedback cancellation, wideband adaptive feedback cancellation techniques do not provide satisfactory performance for reducing feedback oscillation in hearing aids. A band-limited adaptive feedback cancellation algorithm using normalized filtered-X LMS technique provides good cancellation efficiency, convergence behaviour and better output sound quality [16] for speech signals than other wideband method of feedback cancellation.

A generalized noise reduction scheme has been proposed, called the Spatially Pre processed, Speech Distortion Weighted, Multichannel Wiener Filter (SP-SDW-MWF) [17]. It uses Generalized Sidelobe Canceller (GSC) and a multichannel Wiener filtering technique as extreme cases. Compared with the widely studied GSC with Quadratic Inequality Constraint (QIC-GSC), the SP-SDW-MWF achieves a better noise reduction performance for a given maximum speech distortion level. In this paper, a low-cost, stochastic gradient implementation of the SP-SDW-MWF is implemented using frequency domain approach for better speed up convergence and reduces computational complexity. Experimental results with a behind-the-ear hearing aid show that the proposed frequency-domain stochastic gradient algorithm preserves the benefit of the exact SP-SDW-MWF over the QIC-GSC.

For hearing aid users the direct method of adaptive feedback cancellation is widely used to mitigate feedback; however it is less effective for high forward path gains due to the inherent bias in the feedback path estimate. This bias can be reduced at the expense of artificial delays which can potentially introduce pre-echo and “comb filter” effects. The direct method also tends to cancel tonal audio signals such as alarms and music. This study [18] uses closed-loop system identification for unbiased feedback cancellation which does not possess the negative characteristics manifested by the direct method, but uses an identification signal. It uses two-stage method which

employs two adaptive filters, one to identify the entire closed-loop, and another to extract the feedback path response was the preferred unbiased method due to its low computation and good feedback identification particularly during "howling." This paper presents that unbiased feedback cancellation method is better than the widely used direct method.

Stereophonic acoustic echo cancellation has gained much interest in recent years due to the non-uniqueness and misalignment problems that are caused by the strong inter-channel signal coherence. A novel adaptive filtering approach is used to reduce inter-channel coherence which is based on a selective-tap updating procedure [19]. This tap-selection technique is then applied to the normalized least-mean-square, affine projection and recursive least squares algorithms for stereophonic acoustic echo cancellation. This technique reports convergence rate is significantly upon the existing techniques.

An adjustable filter-bank based algorithm has been tested and developed for hearing aid systems [20]. The implementation of this filter-bank algorithm on cochlear-prosthesis' DSP-board enables to generate and control electrical stimulating pulses. In conventional hearing aids driven by DSP, filter bank-based algorithm permits an ease adjustment of speech amplification, which is fully programmable within the considered sounds' spectrum. In each device, a programmable spectrum cut-up permits to adjust filters' bands relatively to patient's pathology. Programming via host computer enables flexibility in speech amplification for conventional hearing aids, and in cochlea's stimulation for cochlear prostheses. It combines handiness, ease of use and safety features to help meet individual's diverse needs. A computer illustration, based on spectrum cutting-up, was designed to identify filters' outputs. Hence, with this visual measure, clinicians could set up experiments for adjusting correctly hearing aid's operation-parameters.

We have already brought out the importance of digital filters, DSP and their importance in Digital hearing aid applications. Furthermore, it is also important that the hearing aid must consume low power and be designed with lowest possible area. The details of low power VLSI design and the approaches adopted in this dissertation will be discussed in the next chapter. Our purpose is to design low power digital filter for hearing aid, we have only chosen suitable algorithms, demonstrated their performance through FPGA implementation.

1.7 Objective of the thesis

From the study of FPGA/ CMOS based techniques of hearing aid; various types of methodologies [11-17] have been applied to hearing aid devices. Some of the circuit level implementation shows the use of direction microphone has been proposed where as others uses switched MOSFET technology to achieve low power. In architectural level of design; use of filtering cores, number presentation like 2's complement and Sign magnitude presentation have been proposed.

In spite of the best efforts, there is still a lot of scope to find and optimize the algorithms suited for hearing aid. However, the algorithm should be hardware implementable as ultimately an IC need to be designed. While designing it is also important to focus on two major issues

a. Low power design
b. Small area.

Keeping in view of above considerations we define the following objectives of the thesis.

a. To investigate and find out suitable algorithm(s) for Hearing aid application with a hardware implementation perspective.
b. To find out an adaptive algorithm that is simple to implement in hardware and also should be less power consuming. Adaptive algorithms are likely to perform better in terms of spectral sharpening and noise reduction. Moreover, since we are attempting for low power design, low power techniques would be adopted at algorithmic abstraction level.
c. Usually some component(s) appear repeatedly in VLSI design. If we can reduce power consumption of such a component at circuit level and incorporate the same in the design; the total power reduction would improve. Therefore, we would proceed to investigate at the circuit level to find a low power circuit that would be extensively used in filter design.
d. To combine circuit level and algorithm level low power consumption in a filter for hearing aid application and to make a real-time validation.
e. To implement for all the filter structures in FPGA.

f. To develop an experimental setup and validate the simulation results and the concepts through experiment for the best structure evaluated in the thesis.

1.8 Organization of thesis

Chapter-I presents introduction to the hearing aid design and basics of literature survey based on algorithms applied to hearing aids.

Chapter-II presents a detailed literature survey low power methodology and FPGA/Circuit level design of digital filters for hearing aids application. Some of the multiplier structures has been studied and simulated.

Chapter-III introduces the use of a Decimation filter for hearing aid application using 'Distributed arithmetic' when the incoming signal is sampled at a higher than Nyquist rate. The filter is being designed using Matlab Simulink and Xilinx System generator. The filter is being tested and the power estimation is carried out.

Chapter-IV introduces the use adaptive filter that incorporates adaptive lattice filter. The filter is being designed using Booth-Wallace multiplier and is implemented in Virtex-II pro board from Xilinx. In this implementation both area and power are computed. Furthermore power computation is also done for the entire hearing aid architecture.

Chapter-V A novel method has been adopted for FIR filter implementation. This filter is being designed and implemented using a MAC unit and FDF (Folded Direct Form) structure for FIR filter. A suitable clocking strategy is adopted which reduces glitches that facilitates reduction of power dissipation. A novel dual edge triggered technique latch together with clock gating strategy is being used to achieve low power and it is observed that this FIR filter in conjunction with clock gating and low power latch consumes low power compared to other designs.

Chapter-VI covers comparison between different types filter and the filter with low power consumption is being implemented with real time signals using Xilinx Virtex-II pro board and spectral analysis is done for three different signals such as speech,

music and voice and music both. The filter successfully works for all these signals. The gain for the filter is being calculated and is found to be between 9 dB to 27 dB which matches specifications given by most of the hearing aid manufacturers.

In **Chapter-VII,** conclusions are brought out and some further research scopes are suggested.

Chapter – II

Low Power VLSI Techniques for Digital Filters for Hearing aid applications

2.1 Review of Low Power Methods used in Filters

The objective of this dissertation is to come up with digital filters that are suitable for hearing aid applications. Here we need to design filters that would provide necessary amplification and frequency response. Furthermore, it is also important that the evolved structures must be hardware realizable and reusable [21], so that we can develop a product. Low power is an essential constraint for hearing aids, therfore suitable low power strategy need to be applied. Low power VLSI techniques at different abstraction levels are applied while designing filter.

Various types of methodologies are employed for low power design. These are

1. Algorithmic level
2. Structural level and
3. Circuit level.

Various filter applications designed at circuit level and FPGA based digital filters also have been studied. Using circuit level design methodologies, an adaptive directional microphone for hearing aid application for DSP VLSI application has been designed with area of 0.67mm^2 based on 250 nm technology with power consumption about 45uW achieved at 1.25V supply [22]. Use of switched MOSFET(SM) technique for low power filter design and a SM programmable band-pass filter for hearing aid application is reported [23]. A high efficiency and low distortion switching power amplifier is proposed and is designed for micropower low-voltage hearing aids. The experimental results show that the proposed circuit has 0.27% total harmonic distortion and 90% power efficiency while the dc output bias current is 19 mA at 1.5V supply voltage which is used in circuit level design for achieving low power [24].

An ultra-low-power delayed least mean square (DLMS) adaptive filter operating in the sub-threshold region for hearing aid applications is reported in which sub-threshold operation was accomplished by using a parallel architecture with pseudo nMOS logic style. The parallel architecture enables to operate the system at a lower clock rate and reduced supply voltage while maintaining the same throughput. Pseudo nMOS logic operating in the sub-threshold region (subpseudo nMOS) provided better power-delay product than sub-threshold CMOS (sub-CMOS) logic [25]. Simulation results show that the DLMS adaptive filter can operate at 22 kHz using a 400-mV supply voltage to achieve 91% improvement in power compared to a nonparallel, CMOS implementation. The result is validated by designing testing of the an carry

save array multiplier test chip with of size a 0.35 m, 23.1 kHz, 21.4 nW, where an adaptive body biasing scheme is used for compensating process, supply and temperature variations.

The following paragraph describes various algorithmic and structural levels to accomplish low power objectives. A multiplier free architecture using Distributed Arithmetic (DA) is used in the implementation of LMS adaptive filter and high speed LMS adaptive filter can be designed using DA in FPGA [26]. Also algorithmic filter cores (structures) for filter implementation has been investigated in order to minimize the switched capacitance of the multipliers and data/coefficient buses [27] and it is reported that a power saving upto 39% can be achieved.

Multipliers play an important role design of filters. A novel design technique for deriving highly efficient multipliers that operate on a limited range of multiplier [28] values is presented in the literature. Using the technique, Xilinx Virtex field programmable gate array (FPGA) implementations for a discrete cosine transform and poly-phase filter were derived with area reductions of 31%–70% and speed increases of 5%–35% when compared to designs using general-purpose multipliers. This design gives better result than other fixed coefficient methods.

Reconfigurable hardware devices make it possible to change structure of digital electronic circuits at runtime. Using reconfigurable devices as a platform for Evolvable hardware (EHW) is well suited for real-time adaptive systems [29] which present both the filter as well as the evolution parameters for adaptive filter implementation on a single Field programmable gate array (FPGA). This paper uses context based switching to obtain a smaller hardware and fast adaption.

Low power block based filtering cores has been studied and specially used for low power filter implementation [30]. This uses algorithmic flow for low power filter design and also uses 2's complement and SM (sign magnitude) number presentation. It has been reported that as compared to convention filtering cores, a power reduction of 49% and area overhead of 5% is increased.

A clock based Low power and area efficient FIR filter of 32 tap using two 16 tap macros implementation has been implemented. Using different conditions for a coded coefficient and data, and a power saving of 35% and 44% improvement as compared to radix-4 modified booth algorithm is being reported [31]. Designs of programmable finite impulse response (FIR) digital filters have demonstrated that the use of broadcast input data and control can lead to a high performance-to-cost ratio. Effect

of the interconnect delay to the cycle time and its negative effects on both scalability and cost-effectiveness of such broadcast designs have been presented in the literature [32]. Further it is shown; speed and density improvements secured through technology scaling can be maintained by a fully pipelined design in which both data and control signals are restricted to local connections [33]. In this design filter coefficients are loaded in bit-parallel form with no increase in the number of input pins, thereby facilitating and speeding up run-time adaptation to the application environment. Another feature of variable-precision coefficients is that, it can be accommodated easily and flexibly, with no speed penalty. This type of design methods can be applied for the design of application- specific and embedded parallel architectures.

There is a continuous drive for methodologies and approaches of low power design. The design of low power systems for different portable applications is not a simple task [33]. This is because of the number of constraints that influence the power consumption of a device. In addition to issues of performance and functionality, there is a need to satisfy strict test coverage constraints. In the same work [33] a brief study on the impact of DSP architectural realization, multiplier type, and the choice of number representation on the overall power consumption of DSP devices have been carried out. Furthermore the effect of DFT circuits on the overall performance has been studied. A hearing aid device is considered as an example of a system with strict power/area constraints. It is being shown that the choice of multiplier architecture and number representation should be carefully considered when specific DSP architectural choices are made.

The design and implementation of decimation filter used for hearing aid applications is reported in [33]. The implementation of the decimation filters using the canonical signed digit (CSD) representation is being carried out [34]. Each digital filter structure is simulated using Matlab, and its complete architecture is captured using DSP blockset and Simulink. The filter has been implemented on Xilinx FPGA using Virtex-II technology. The resulting architecture is hardware efficient and consumes less power compared to conventional decimation filters [34]. Compared to the comb-FIR-FIR architecture, the designed decimation filter architecture contributes to a hardware saving of 69%; in addition, it reduces the power dissipation by 83%, respectively. In another work a very fast and low complexity FIR filter implemented using CSD multiplication is realized using shifters, adders and subtractors [35]. In this method, the critical path is minimized using pipeline registers equal to propagation

delay of an adder. For 100% speed up an area overhead of 5% increase has been reported.

Another work describes a scheme for the implementation of low power cores for hearing aid applications [36]. In this work, power saving schemes has been investigated using two approaches. First method uses macro-component framework which allows the rapid assembly of the cores on easy-to-verify hierarchical plug-in basis and second one uses system-on-chip strategy [36]. The cores are embedded within an ARM-based system-on-chip platform for design and testing of FIR filters. Using this method, it is shown that effective power manipulation is possible by power management strategies.

The design and implementation of an improved hardware-based evolutionary digital filter (EDF) version 2 is being reported in [37]. EDF is an adaptive digital filter which is controlled by adaptive algorithm based on evolutionary computation. The hardware based EDF version 1 consists of two sub-modules, that is, a filtering and fitness calculation (FFC) module and a reproduction and selection (RS) module. The FFC module has high computational ability to calculate the output and the fitness value since its sub-modules run in parallel. However, hardware size of the FFC module is large, and many machine cycles are needed. Thus, in the hardware-based EDF version 2 combines two modules to reduce its hardware size and machine cycles [36]. A synthesis result on the FPGA shows that the clock frequency is 65.5MHz and the maximum sampling rate of the hardware-based EDF version 2 is 4,948.1Hz. Moreover, the hardware-based EDF version 2 is 15.7 times faster than the hardware-based EDF version 1.

A coefficient segmentation algorithm for low power FIR filter implementation has been proposed and the algorithm decomposes individual coefficients into two primitive sub-components [38]. The decomposition, performed using a heuristic approach, divides a given coefficient such that a part is produced which can be implemented using a single shift operation leaving another part with a reduced word length to be applied to the coefficient input of the hardware multiplier. This results in a significant reduction in the amount of switched capacitance and consequently power consumption. The algorithm has been used with a number of practical FIR filter examples achieving up to 63% saving in power.

Another paper describes the design of a compact, ultra low power continuous time programmable filter suitable for feedback cancellation filters in hearing aids and

open loop identification is being performed [39]. Because of the complexity of the transfer function the number of poles in the cancellation filter is fairly large, ultra-low power transconductance cell with large linear range has been designed. The power consumption for a transconductance cell is comparable to the theoretical minimum bound and is measured to be 0.8μW at a 3V supply has been reported. The linear input range for this cell is 2Vp-p. The complete filter has been designed, implemented and fabricated in a 2μ CMOS technology.

In spite of all the cited work [21-39] there is still a lot of scope for alternate designs at different abstraction levels on low power perspective. Therefore an attempt is being made to design digital filters that are not only functionally suitable, but those would operate at low power that is one of the primary constraints for hearing aid design.

In order to proceed further, we now discuss the basics of low power VLSI design. This study would provide us greater insight for low power VLSI design of digital filters.

2.2 Basics of Low power VLSI Design

In the past few years there has been an explosive growth in the demand for portable computing and communication devices, from mobile telephones to sophisticated multimedia systems [40]. This interest in these devices has enhanced the requirement of developing low-power signal processors and algorithms, as well as the development of low-power general purpose processors. Designers have been able to reduce the energy requirements of particular functions, such as video compression, by several orders of magnitude [41]. This reduction has come as a result of focusing on the power dissipation at all levels of the design process, from algorithm design to the detailed implementation, however, there has been little work done to understand how to design energy efficient processors.

Performance of processors has been growing at an exponential rate, doubling every 18 to 24 months. However, at the same time the power dissipated by these processors has also been growing considerably. For such processors cooling becomes an absolute necessity and at high power dissipation level this is even difficult and expensive. If this trend continues processors will soon dissipate hundreds of watts, which would be unacceptable in most systems. Thus there is great interest in under-

standing how to continue increasing performance without increasing the power dissipation.

For portable applications the problem is even more severe since battery life depends on the power dissipation. Lithium-ion batteries have an energy density of approximately 100Wh/kg, the highest available today. To operate a 50 W processor for 4 hours requires a 2 kg battery; hence it can hardly be termed as a portable device. In order to compare processor designs that have different performance and power one needs a measure of "goodness". If two processors have the same performance or the same power, then it is trivial to choose which is better—users prefer higher performance for the same power level or the lower power one if they have the same performance. But processor designs rarely have the same performance. Designers have to determine whether to add a particular feature will make a processor more desirable or not. However micro-architectural designing changes the amount of parallelism of the processor, affects the efficiency of the processor. Since both the performance and energy dissipation of modern processors depend heavily on the design of the memory hierarchy, one must look not only at the processor itself, but also have to look at the design of the memory. Since memories and clocking circuits are critical components of every digital system, much work already has been done to reduce the energy requirements. A different approach to reduce the energy dissipation of clocks and memories is to change the technology by scaling the supply voltage and the threshold voltage of transistors.

2.3 Power Dissipation Sources

In CMOS circuits, the main contributions to the power consumption are from short-circuiting current, leakage current, and switching currents [45], [49]. In the following subsections, we introduce them separately.

2.3.1 Short-Circuit Power

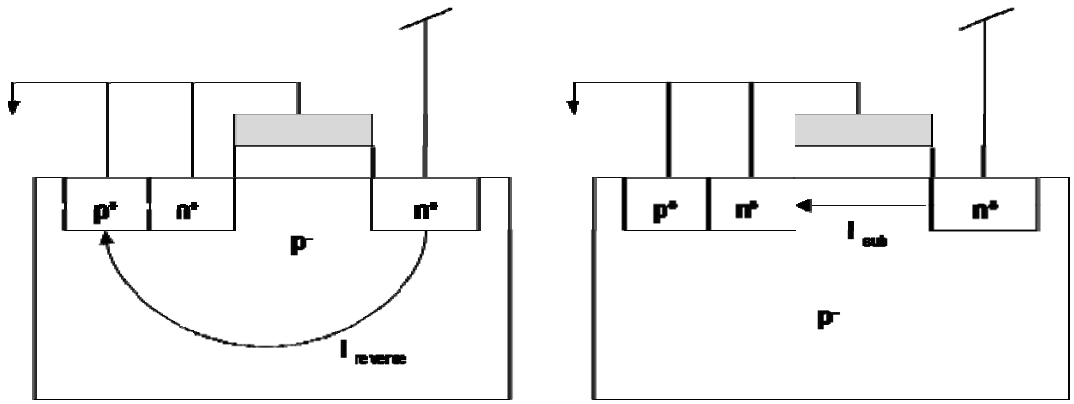
In a static CMOS circuit, there are two complementary networks: p-network (pull-up network) and n-network (pull-down network). The logic functions for the two networks are complementary to each other. Normally when the input and output state are stable, only one network is turned on and conducts the output either to power supply node or to ground node and the other network is turned off and blocks the

current from flowing. Short-circuit current exists during the transitions as one network is turned on and the other network is still active. For example, the input signal to an inverter is switching from 0 to V_{dd}. During this transaction, there exists a short time interval where the input voltage is larger than V_{tn} but less than $V_{dd} - |V_{tp}|$. During this time interval, both PMOS-transistor (p-network) and NMOS-transistor (n-network) are turned on and short-circuit current flows through both kinds of transistors from power supply line to the ground.

The exact analysis of the short-circuit current in a simple inverter [43] is complex; this is analyzed by SPICE simulation. It is observed that the short-circuit current is proportional to the slope of input signals, the output loads and the transistor sizes. The short-circuit current consumes typically less than 10% of the total power in a "well-designed" circuit [46].

2.3.2 Leakage Power

Leakage currents are due to two sources: one from the currents that flow through the reverse biased diodes (reverse biased PN-Junction current), the other from the currents that flow through transistors that are non-conducting (sub-threshold channel conduction current) as shown in figure 2.1.



(a) (b)

Figure 2. 1: Leakage current types: (a) reverse biased diode current, (b) sub-threshold leakage current.

The leakage currents are proportional to the leakage area and exponential of the threshold voltage. The leakage currents are due to manufacturing technology and cannot be modified by the designers except in some logic styles. Sub-threshold leak-

age and reverse-biased junction leakage current; both increases dramatically with temperature and are independent of the operating voltage for a given fabrication process.

The leakage current is in the order of pico-Ampere, but it increases as the threshold voltage is reduced. In some cases, like large RAMs, the leakage current is one of the main concerns. The leakage current is currently not a severe problem in most digital designs. However, the power consumed by leakage current can be as large as the power consumed by the switching current for $0.06\mu m$ technology. The usage of multiple threshold voltages can reduce the leakage current in deep-submicron technology. Leakage current is difficult to predict, measure or optimized. Generally, leakage current serves no useful purposes, but some circuits do exploit it for intended operations, such as power-on reset signal generation. The leakage power problem mainly appears in very low frequency circuits or ones with "sleep modes" where dynamic activities are suppressed.

2.3.3 Switching Power

The switching currents are due to the charging and discharging of node capacitances. The node capacitances mainly include gate, overlapping, and interconnection capacitances. The power consumed by switching [43] current can be expressed as

$$P = \alpha C_L f V_{dd}^2 / 2 \qquad (2.1)$$

where α a is the switching activity factor, C_L is the load capacitance, f is the clock frequency, and V_{dd} is the supply voltage.

The above equation (2.1) shows that the switching power depends on a few quantities that are readily observable and measurable in CMOS circuits. It is applicable to almost every digital circuit and hence provides guidelines for the low power design.

The power consumed by switching current is the dominant part of the power consumption. Reducing the switching current is the focus of most low power design techniques. For large capacitance circuits, reduction of the frequency is the best way to reduce the switching power. The use of different coding methods, number repre-

sentation systems, continuing sequences and data representations can directly alter the switching frequency of the design, which alters the switching power. The best method of reducing switching frequency is to eliminate logic switching that is not necessary for computation.

2.4 Low Power Techniques

Low power techniques can be discussed at various levels of abstractions: system level [43] [46], algorithm and architecture level, logic level, circuit level, and technology level. Figure 2.2 shows some examples of techniques at the different levels.

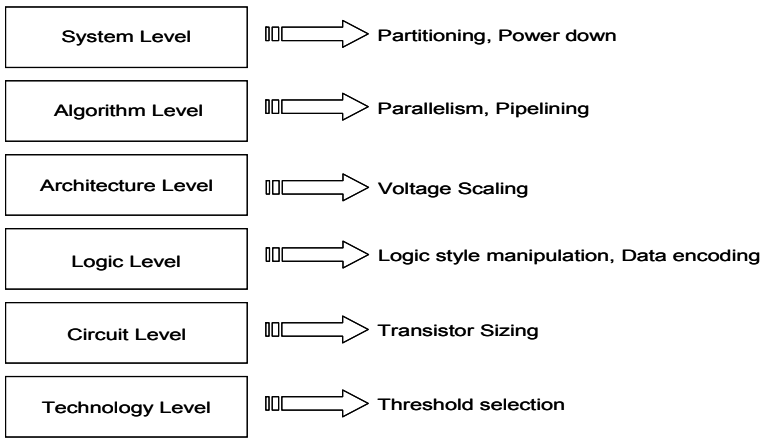


Figure 2. 2: Low-power design methodology at different abstraction levels.

In the following sections, an overview for different low power techniques has been described in detail. This is organized on the basis of abstraction level.

2.4.1 System Level

A system typically consists of both hardware and software components, which affect the power consumption. The system design includes the hardware/software partitioning, hardware platform selection (application-specific or general-purpose processors), resource sharing (scheduling) strategy, etc. The system design usually has the largest impact on the power consumption and hence the low power techniques applied at this level have the most potential for power reduction.

At the system level, it is hard to find the best solution for low power in the large design space and there is a shortage of accurate power analysis tools at this level. However, if, for example, the instruction-level power models for a given processor are

available, software power optimization can be performed [50]. It is observed that faster code and frequent usage of cache are most likely to reduce the power consumption. The order of instructions also have an impact on the internal switching within processors and hence on the power consumption.

The power-down and clock gating are two of the most used low power techniques at system level. The non-active hardware units are shut down to save the power. The clock drivers, which often consume 30-40% of the total power consumption, can be gated to reduce switching activities as illustrated in figure 2.3.

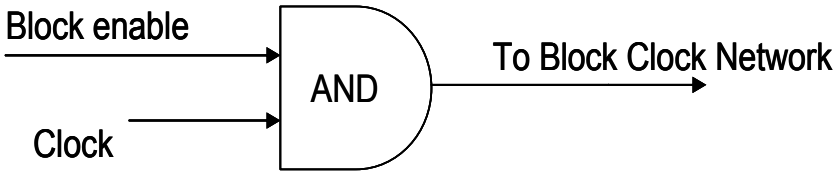


Figure 2. 3: Clock gating.

The power-down can be extended to the whole system. This is called sleep mode and widely used in low power processors. The system is designed for the peak performance. However, the computation requirement is time varying. Adapting clocking frequency and/or dynamic voltage scaling to match the performance constraints is another low power technique. The lower requirement for performance at certain time interval can be used to reduce the power supply voltage. This requires either feedback mechanism (load monitoring and voltage control) or predetermined timing to activate the voltage down-scaling.

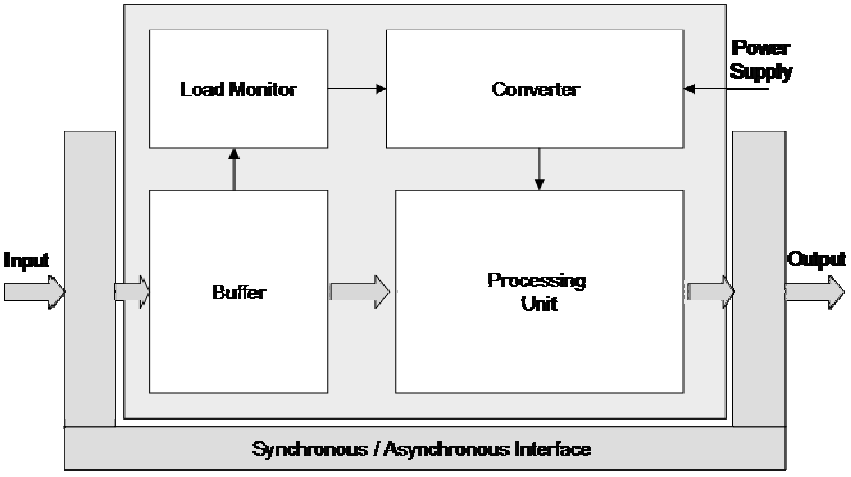


Figure 2. 4: Asynchronous design with dynamic voltage scaling.

Asynchronous design of the circuit can also be used as another low power designing technique. The asynchronous designs have many attractive features, like non-global clocking, automatic power-down, no spurious transitions, and low peak current, etc. It is easy to reduce the power consumption further by combining the asynchronous design technique with other low power techniques, for instance, dynamic voltage scaling [47] technique as shown in figure 2.4.

2.4.2 Algorithm Level

The algorithm selection has large impact on the power consumption. The task of algorithm design is to select the most energy-efficient algorithm that just satisfies the constraints. The cost of an algorithm includes the computation part and the communication/storage part. The complexity measurement for an algorithm includes the number of operations and the cost of communication and storage. Reduction of the number of operations, cost per operation, and long distance communications are key issues to algorithm selection.

One important technique for low power of the algorithmic level is algorithmic transformations [48]. This technique exploits the complexity, concurrency, regularity, and locality of an algorithm. Reducing the complexity of an algorithm reduces the number of operations and hence the power consumption. The possibility of increasing concurrency in an algorithm allows the use of other techniques, e.g., voltage scaling, to reduce the power consumption. The regularity and locality of an algorithm affects the controls and communications in the hardware.

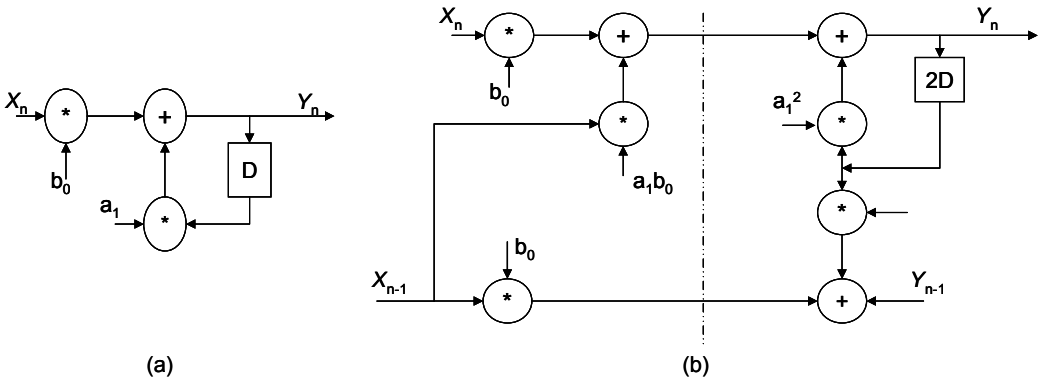


Figure 2. 5: (a) Original signal flow graph. (b) Unrolled signal flow graph.

The loop unrolling technique [41-42, 45] is a transformation that aims to enhance the speed. This technique can be used for reducing the power consumption. With loop unrolling, the critical path can be reduced and hence voltage scaling can be applied to reduce the power consumption.

In figure 2.5, the unrolling reduces the critical path and gives a voltage reduction of 26% [43]. This reduces the power consumption with 20% even the capacitance load is increased to 50% [41]. Furthermore, this technique can be combined with other techniques at architectural level, for instance, pipeline and interleaving, to save more power. In some cases, like digital filters, the faster algorithms, combined with voltage-scaling, can be used for energy-efficient applications [43].

2.4.3 Architecture Level

According to the selection of the algorithm, the architecture can be determined for the given algorithm. From equation (1) we can say that, an efficient way to reduce the dynamic power consumption is the voltage scaling. When supply voltage is reduced, the power consumption is reduced. However, this increases the gate delay. To compensate the delay, low power techniques like parallelism and pipelining [44] architectures were used.

The use of two parallel data path is equivalent to interleaving of two computational tasks. A data path to determine the largest number of C and (A + B) is shown in figure 2.6. It requires an adder and a comparator. The original clock frequency is 40 MHz [42].

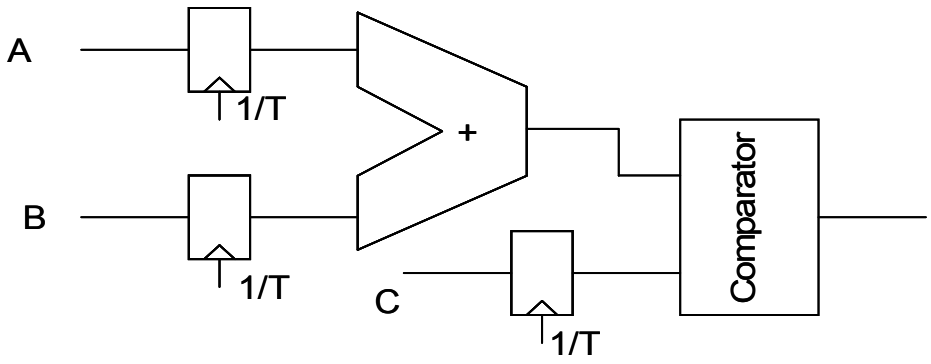


Figure 2. 6: Original data path.

In order to maintain the throughput while reducing the power supply voltage, we use a parallel architecture. The parallel architecture with twice the amount of resources is shown in figure 2.7. The clock frequency can be reduced to half, from 40 MHz to 20 MHz since two tasks are executed concurrently. This allows the supply voltage to be scaled down from 5 V to 2.9 V [44]. Since the extra routing is required to distribute computations to two parallel units, the capacitance load is increased by a factor of 2.15. The power is calculated by using equation 2.2.

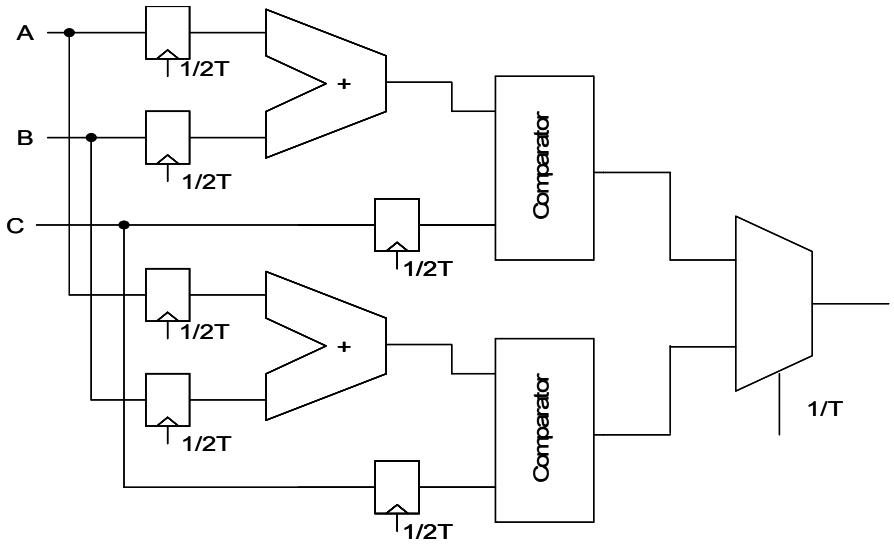


Figure 2. 7: Parallel implementation.

$$P_{par} = C_{par} V_{par}^2 f_{par} = (2.15 C_{actual})(0.58 V_{actual})^2 (\frac{f_{actual}}{2}) = 0.36 P_{actual} \qquad (2.2)$$

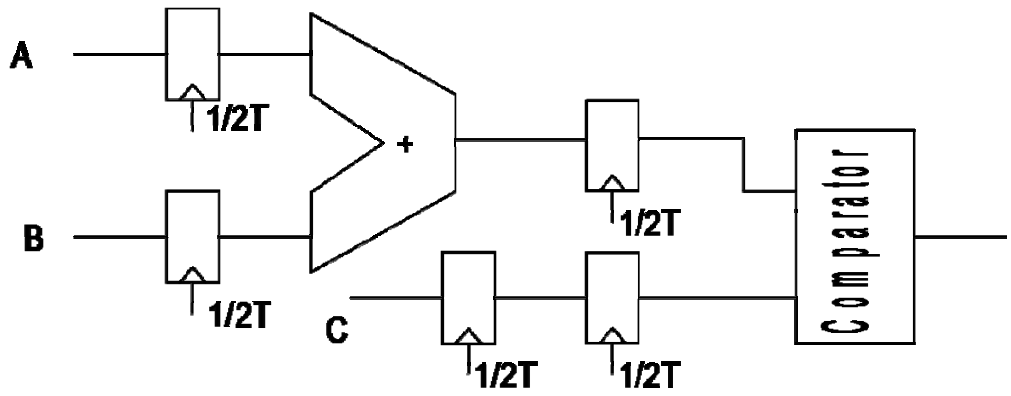


Figure 2. 8: Pipelining implementation.

$$P_{pipe} = C_{pipe} V_{pipe}^2 f_{pipe} = (1.15 C_{actual})(0.58 V_{actual})^2 (f_{actual}) = 0.39 P_{actual} \qquad (2.3)$$

Pipelining is another method for increasing the throughput. By adding a pipelining buffer / register after the adder in figure 2.8., the throughput can the increased from $1/(T_{add} + T_{comp})$ to $1/\max(T_{add}, T_{comp})$. If T_{add} is equal T_{comp}, this increases the throughput by a factor of 2. As a result the supply voltage also scaled down to 2.9 V (the gate delay doubles). The effective capacitance increases to a factor of 1.15 because of the insertions of latches. The power consumption for pipelining is calculated using equation (2.3).

Main advantage of pipelining is the low area overhead in comparison with using parallel data paths. Another benefit is that the amount of glitches can be reduced. However, since the delay increases significantly as the voltage approaches the threshold voltage and the capacitance load for routing and/or pipeline registers increases, there exists an optimal power supply voltage. Reduction of supply voltage lower than the optimal voltage increases the power consumption.

2.4.4 Logic Level

The power consumption depends on the switching activity factor, which in turn depends on the statistical characteristics of data. The low power techniques at the logic level, however, focus mainly on the reduction of switching activity factor by using the signal correlation and the node capacitances. In case of the gated clocking, the clock input to non-active functional block does not change by gating, and, hence, reduces the switching of clock network.

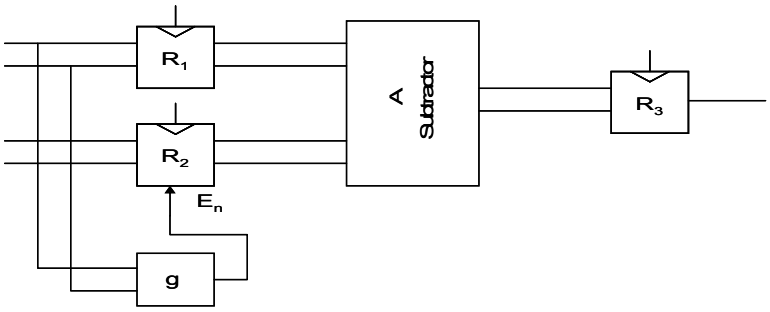


Figure 2. 9: A pre-computation structure for low power.

Pre-computation [43] uses the same concept to reduce the switching activity factor: a selective pre-computing of the output of a circuit is done before the outputs are required, and this reduces the switching activity by gating those inputs to the circuit.

As shown in figure 2.9, the input data is partitioned into two parts, corresponding to registers R_1 and R_2. One part, R_1, is computed in pre-computation block g, one clock cycle before the computation of A. The result from g decides gating of R_2. The power can then be saved by reducing the switching activity factor in A.

Let's consider a comparator as an example of pre-computation for low-power. The comparator takes the MSB of the two numbers to register R_1 and the others to R_2. The comparison of MSB is performed in g. If two MSBs are not equal, the output from g gated the remaining inputs. In this way, only a small portion of inputs to the comparator's main block A (subtracter) is changed. Therefore the switching activity is reduced.

Gate reorganization [43] is another technique used to restructure the circuit. This can be decomposition a complex gate to simple gates, or combines simple gates to a complex gate, duplication of a gate, deleting/addition of wires. The decomposition of a complex gate and duplication of a gate help to separate the critical and non-critical path; which reduce the size of gates in the non-critical path, as a result reduces the power consumption. In some cases, the decomposition of a complex gate increases the circuit speed and gives more space for power supply voltage scaling. The composition of simple gates can reduce the power consumption. The complex gate can reduce the charge/discharge of high-frequency switching node. The deleting of wires reduces the circuit size as a result, reduces the load capacitance. The addition of wires helps to provide an additional interconnection for better results.

Logic encoding defines the way data bits are represented on the circuits. The encoding is usually optimized for reduction of delay or area. In low power design, the encoding is optimized for reduction of switching activities since various encoding schemes have different switching properties.

In a counter design, counters with binary and Gray code have the same functionality. For N-bit counter with binary code, a full counting cycle requires $2(2^n - 1)$ transitions [45]. A full counting cycle for a Gray coded N-bit counter requires only 2^n transitions. For instance, the full counting cycle for a 2-bit binary coded counter is from 00, 01, 10, 11, and back to 00, which requires 6 transitions. The full counting cycle for 2-bit Gray coded counter is from 00, 01, 11, 10, and back to 00, which requires

4 transitions. The binary coded counter has twice transitions as the Gray coded counter when the n is large. Using binary coded counter therefore requires more power consumption than using Gray coded counter under the same conditions.

Traditionally, the logic coding style is used for enhancement of speed performance. Careful choice of coding style is important to meet the speed requirement and minimize the power consumption. This can be applied to the finite state machine, where states can be coded with different schemes.

A bus is the main on-chip communication channel that has large capacitance. As the on-chip transfer rate, increases the use of buses contributes a significant portion of the total power. Bus encoding is a technique to exploit the property of transmitted signal to reduce the power consumption.

2.4.5 Circuit Level

At the circuit level, the powers saving techniques are quite limited if compared with the other techniques at higher abstract levels. However, this cannot be ignored. The power savings can be significant as the basic cells are frequently used. A few percents improvement for D flip-flop can significantly reduce the power consumption in deep pipelined memory systems.

In CMOS circuits, the dynamic power consumption is caused by the transitions. Spurious transitions typically consume between 10% and 40% of the switching activity power in the typical combinational logic. In some cases, like array multipliers, the amount of spurious transitions is large. To reduce the spurious transitions, the delays of signals from registers that converge at a gate should be roughly equal. This can be done by insertions of buffers and device sizing [43]. The insertions of buffer increase the total load capacitance but can still reduce the spurious transitions. This technique is called path balancing.

Many logic gates have inputs that are logically equivalent, i.e., the swapping of inputs does not modify the logic function of the gate. Some examples of gates are NAND, NOR, XOR, etc. However, from the power consumption point of view, the order of inputs does effect the power consumption. Let us consider the figure 2.10, the A-input, which is near the output in a two-input NAND gate, consumes less power

than the B-input close to the ground with the same switching activity factor. Pin ordering is to assign more frequently switching input pins near to the output node, which will consume less power. In this way, the power consumption will be reduced without cost. However, the statistics of switching activity factors for different pins must be known in advanced and this limits the use of pin ordering [45].

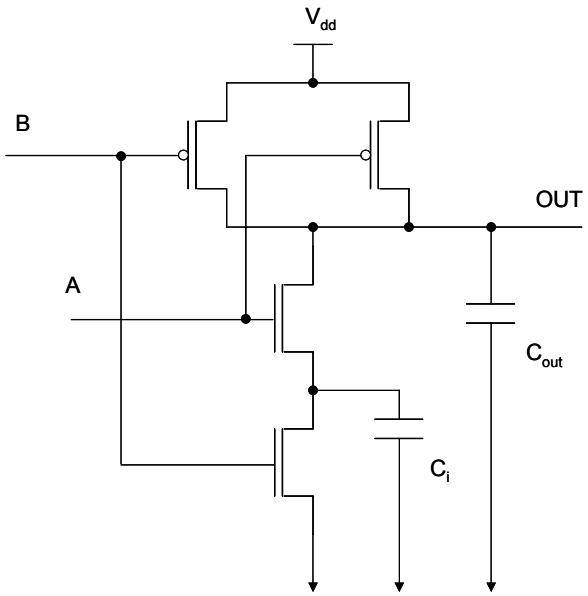


Figure 2. 10: A two input NAND gate.

Different logic styles have different electrical characteristics. The selection of logic style affects the speed and power consumption. In most cases, the standard CMOS logic is used for speed and power trade-off. In some cases other logic styles, like complementary pass-transistor logic (CPL) is efficient.

Transistor sizing affects both delay and power consumption. Generally, a gate with smaller size has smaller capacitance and consumes less power. To minimize the transistor sizes and meet the speed requirement is a trade-off. Typically, the transistor sizing uses static timing analysis to find out those gates (whose slack time is larger than 0) to be reduced. The transistor sizing is generally applicable for different technologies.

2.4.6 Summary of low power VLSI design technology

Several approaches to reduce the power consumption have been briefly discussed. Below we summarize some of the most commonly used low power techniques.

- Reduce the number of operations. The selection of algorithm and/or architecture has significant impact on the power consumption.

- Power supply voltage scaling. The voltage scaling is an efficient way to reduce the power consumption. Since the throughput is reduced as the voltage is reduced, this may need to be compensated by using parallel and/or pipelining techniques.

- I/Os between chips can consume large power due to the large capacitive loads. Reducing the number of chips is a promising approach to reduce the power consumption.

- Power management. In many systems, the most power consuming parts are often idle. For example, in a lap-top computer, the portion of display and hard disk could consume more than 50% of the total power consumption. Using power management strategies to shut down these components when they are idle for a long time can achieve good power saving.

- Reducing the effective capacitance. The effective capacitance can be reduced by several approaches, for example, compact layout and efficient logic style.

- Reduce the number of transitions. To minimize the number of transitions, especially the glitches, is important

2.5 Multiplier structure review

Multiplier is an important component in a digital filter. In the next sections we review multipliers structures and present some simulation results.

2.5.1 Introduction

The multiplier is an important part of digital signal processors (DSP) because it typically determines the performance of the chips. Furthermore, for a highly complex circuit, the power consumption and the layout area are the two design considerations of the multiplier. In some of the DSP applications, precision can he sacrificed to im-

prove the speed and to reduce the area. Therefore, several fixed-width or reduced-width multipliers have been proposed for this purpose.

To save significant power consumption, it is a good to reduce dynamic power which is the major part of total power dissipation. Besides, a functional unit that has versatile functionalities is much helpful to increase the flexibility of the encapsulated processing core in the portable devices.

A number of studies are undertaken to reduce the dynamic power by minimizing the switching capacitance, which are analysed in details in [51]. In addition, it is shown that the design of a multiplier based on Booth algorithm along with a Dynamic-Range Determination (DRD) unit to select the input operand with a smaller effective dynamic range reduces power dissipation. However, the DRD unit induces additional delay and area overheads, and the input data flows are also switched frequently when the smaller effective dynamic range input data often change from operand A to operand B, and vice versa. In such cases, the power dissipation of the design [52] will be increased rather than decreased. The design of above multipliers uses latches to synchronize the inputs to the adders in the adder tree of multipliers. The asserting signal is propagated by a series of delay circuits to each adder in the adder tree, which induces significant delay cost to the multipliers. Hence, the multipliers presented in [53] are restricted to low-speed applications.

2.5.2 Background of Multipliers

2.5.2.1 Basic binary multiplier

The shift-add Multiplier scheme is the most basic of unsigned Integer multiplication algorithms. The operation of multiplication is rather simple in digital electronics. It has its origin from the classical algorithm for the product of two binary numbers. This algorithm uses addition and shift left operations to calculate the product of two numbers. Two examples are presented below in figure 2.10.

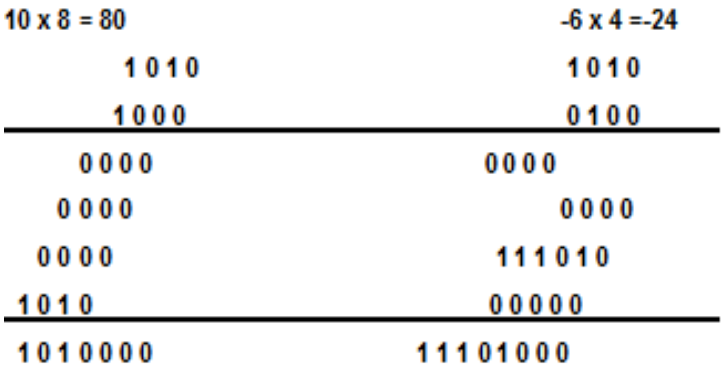


Figure 2. 11: Basic binary multiplication

The left example shows the multiplication procedure of two unsigned binary digits while the one on the right is for signed multiplication. The first digit is called Multiplicand and the second Multiplier. The only difference between signed and unsigned multiplication is that we have to extend the sign bit in the case of signed one, as depicted in the given right example in PP row 3. Based upon the above procedure, we can deduce an algorithm for any kind of multiplication which is shown in figure 2.12. Here, we assume that the MSB represents the sign of digit.

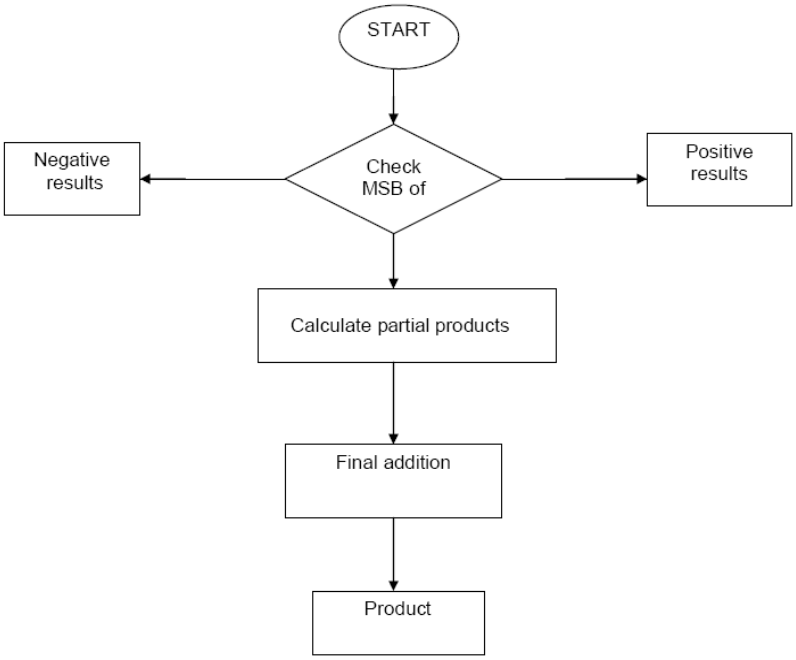


Figure 2. 12: Signed multiplication algorithm

2.5.2.2 Partial product generation

Partial product generation is the very first step in binary multiplier. These are the intermediate terms which are generated based on the value of multiplier. If the multiplier bit is '0', then partial product row is also zero, and if it is '1', then the mul-

tiplicand is copied as it is. From the 2nd bit multiplication onwards, each partial product row is shifted one unit to the left as shown in the above mentioned example. In signed multiplication, the sign bit is also extended to the left. Partial product generators for a conventional multiplier consist of a series of logic AND gates as shown in Figure 2.13.

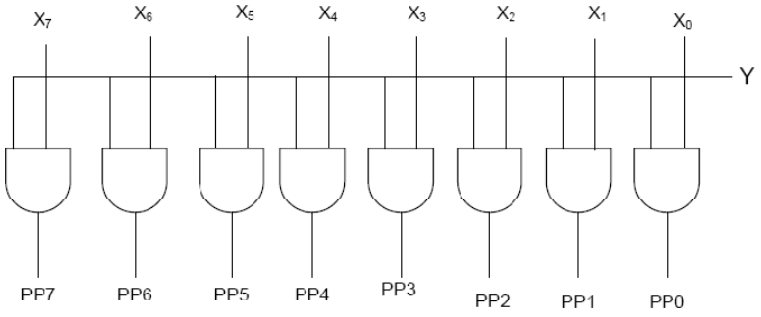


Figure 2. 13: Partial product generation logic

The main operation in the process of multiplication of two numbers is addition of the partial products. Therefore, the performance and speed of the multiplier depends on the performance of the adder that forms the core of the multiplier. To achieve higher performance, the multiplier must be pipelined. Throughput is often more critical than the cycle response in DSP designs. In this case, latency in the multiply operation is the price for a faster clock rate.

This is accomplished in a multiplier by breaking the carry chain and inserting flip-flops at strategic locations. Care must be taken that all inputs to the adder are created by signals at the same stage of the pipeline. Delay at this point is referred to as latency.

2.5.2.3 Introduction of Tree Structured Multiplier

In conventional RL linear array multipliers, the PPs are added in series starting from *y*0*X* (*z*0*X* in radix-4), as shown in Figure 2.14a, 2.15a. The reduction is usually performed using [3:2] Carry Save Adders (CSAs) in which carries pass to the next row. A radix-2 8 × 8 RL multiplier is depicted in figure 2.16. The black dots correspond to the bit matrix in Figure 4a, obtained with NAND2 or AND2 gates. Each '+' symbol is a full adder (FA) if there are three inputs, or a half adder (HA) if there are only two inputs [51][56]. The '1' in the first row in Figure 4a is added as a carry-in of the final Carry Propagate Adders (CPA). The numbers associated with wires are signal arrival

times assuming a unit delay model as explained later. The final addition is a fast *n*-bit CPA.

In LR linear array multipliers, the PPs are added serially from y_n-1X or $z_n/2-1X$, as shown in Figure 2.14b and 1.15b. PP reduction using CSAs is still a popular choice here. For radix-2, a 12 × 12 LR multiplier architecture using CSAs is illustrated in Figure 7. The final (n − 1)-bit CPA generates the most- significant half of the product.

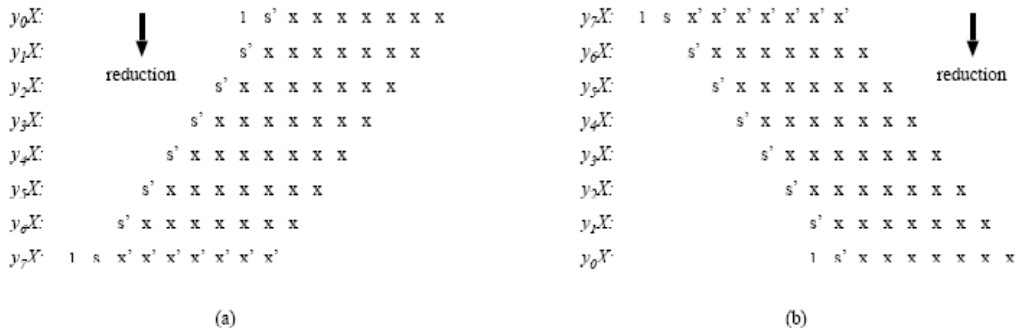


Figure 2. 14: Radix-2 PP bit matrix(n=8): (a) RL; (b) LR

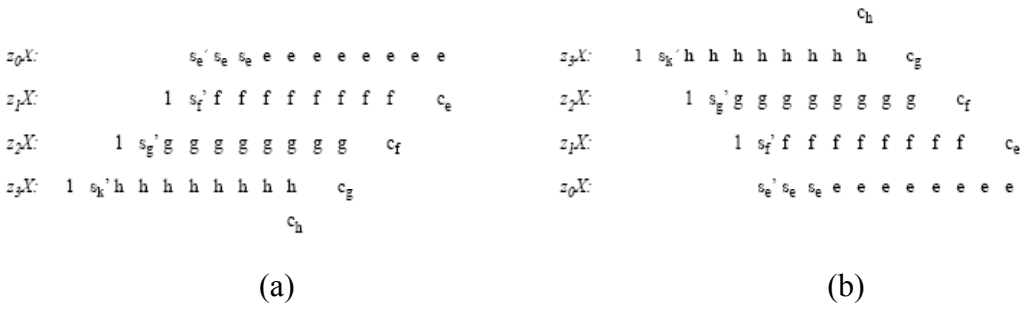


Figure 2. 15: Radix-4 PP bit matrix(n=8) : (a) RL;(b) LR

The shaded cells in the last row comprise a CRA which generates the least-significant half of the product and a carry-in of the final CPA. As the arrival times of these carry/sum bits match the computation direction and speed of the Carry Ripple Adders (CRA), there is little delay penalty due to the use of CRA. The shaded cells on the left are used to add three bits each column from the reduction array into two bits. These shaded cells are extra hardware unique to LR multipliers. However, these extra cells do not increase the overall area because the number of cells in the Partial Product Reduction (PPR) core is reduced.

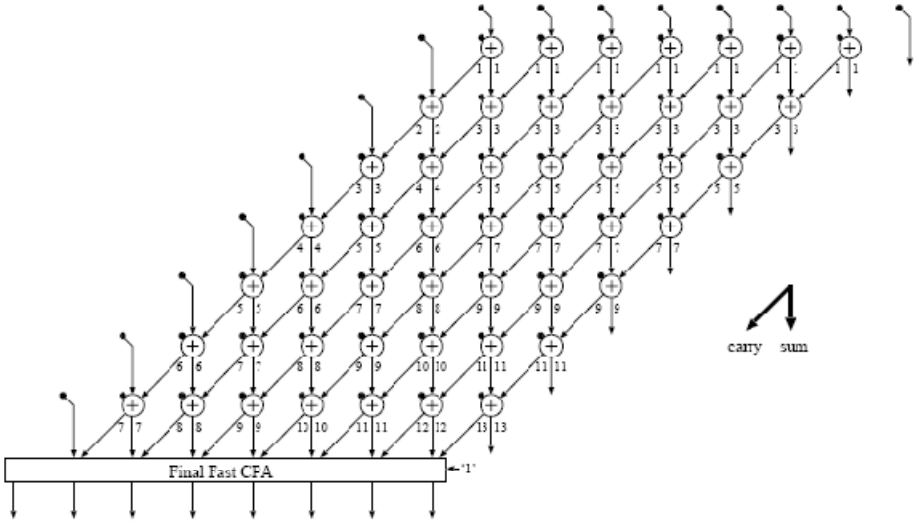


Figure 2. 16: A radix-2 8 × 8 RL multiplier

The carry signals propagate fewer stages in RL schemes than in RL schemes, which may reduce the power consumption in the left region. For data with a large dynamic range, PPs corresponding to sign extension bits are located in the upper region of an LR array. If sign extension bits switch less frequently than other bits, as in many multimedia data, glitches can be reduced because the upper portion is not polluted by frequent switches in the lower portion in LR arrangement. In radix-4 LR multipliers, the CRA is no longer suitable to add the right half carry/sum vectors from the reduction array because the vector bits arrive faster than the CRA computation. To avoid becoming the critical path, CRA should be replaced by a fast CPA as shown in figure 2.17.

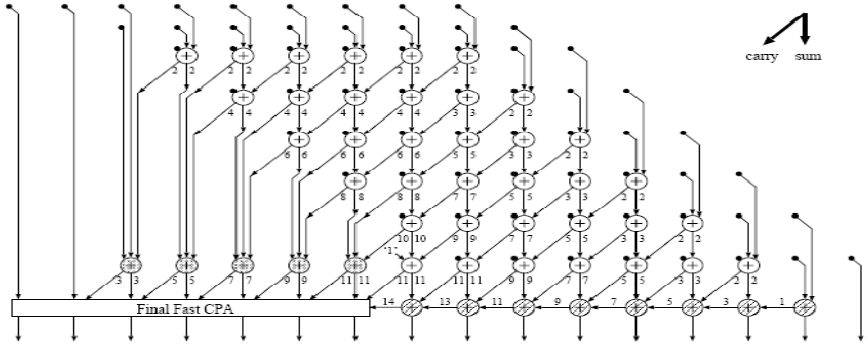


Figure 2. 17; Radix-2 LR carry save array multiplier (n=8)

For vectors from the left part of the reduction array, CSAs are still needed because about half columns have three bits. The total area of a radix-4 LR multiplier is also very close to that of a radix-4 RL multiplier. A 12 × 12 multiplication example is shown in figure 2.18.

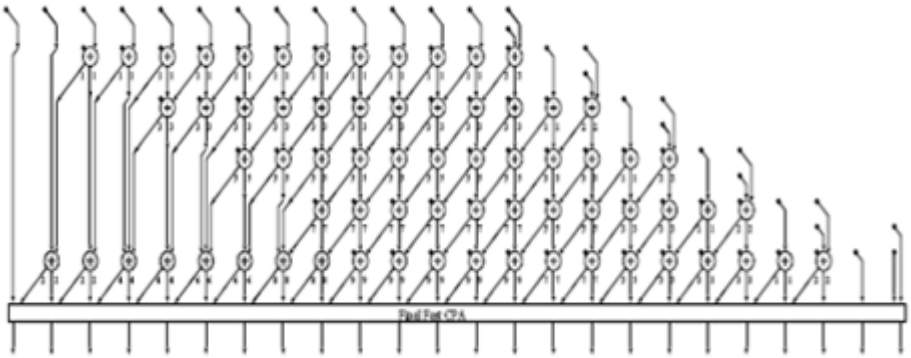


Figure 2. 18: Radix-2 LR carry save array multiplier (n=12)

2.5.3 Speeding Up of Multiplication

Multiplication involves two basic operations - generation of partial products and their accumulation. Two ways to speed up multiplication

a. Reducing number of partial products and/or
b. Accelerating accumulation

2.5.3.1 Sequential multiplier

Generates partial products sequentially and adds each newly generated product to previously accumulated partial product. Example: add and shift method. The following notation is used in our discussion of multiplication algorithms:

a Multiplicand a(k-1)a(k-2) ... a(1)a(0)

x Multiplier x(k-1)x(k-2) ... x(1)x(0)

P Product (a x x) a(2k-1)a(2k-2) ... a(1)a(0)

Sequential or bit-at-a-time multiplication can be done by keeping a cumulative partial product (initialized to 0) and successively adding to it the properly shifted terms x(j)a. Since each successive number to be added to the cumulative partial product is shifted by one bit with respect to the preceding one, a simpler approach is to shift the cumulative partial product by one bit in order to align its bits with those of the next partial product.

2.5.3.2 Parallel multiplier

Generates partial products in parallel, accumulates using a fast multi-operand adder. Number of partial products can be reduced by examining two or more bits of a multiplier at a time. Example: Booth's algorithm reduces number of multiplications to n/2 Where n is the total number of bits in a multiplier [57].

2.5.3.3 Booth's Multiplier

In add and shift algorithm the initial partial product is taken as zero. In each step of the algorithm, LSB bit of the multiplier is tested, discarding the bit which was previously tested, and hence generating the individual partial products. These partial products are shifted and added at each step and the final product is obtained after n steps for n x n multiplication. The main disadvantage of this algorithm is that it can be used only for unsigned numbers. The range of the input for a 'n' bit multiplication is from 0 to 2n-1. A better algorithm which handles both signed and unsigned integers uniformly is Booth's algorithm. Booth encoding is a method used for the reduction of the number of partial products proposed by A.D. Booth in 1950 [57-59].

$$X=-2^{m}X_{m} + 2^{m-1} X_{m-1} + 2^{m-2} X_{m-2}+\ldots\ldots$$

Rewriting above equation using $2^{a}=2^{a+1} - 2^{a}$ leads to

$$X=-2^{m}(X_{m-1} -X_{m})+2^{m-1}(X_{m-2}+X_{m-1})+2^{m-2}(X_{m-3}-X_{m-2})$$

Considering the first 3 bits of X, we can determine whether to add Y, 2Y or 0 to partial product. The grouping of X bits is shown in figure 2.19

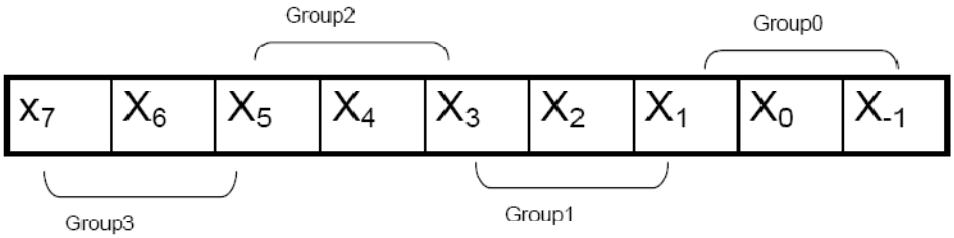


Figure 2. 19: Multiplier bit grouping according to Booth encoding

The multiplier X is segmented into groups of three bits (X_{i+1}, X_i, X_{i-1}) and each group of bits is associated with its own partial product row using table 2.1.

Table 2. 1: Booth encoding table

X_{i+1}	X_i	X_{i-1}	Increment
0	0	0	0
0	0	1	Y
0	1	0	Y
0	1	1	2Y
1	0	0	-2Y
1	0	1	-Y
1	1	0	-Y
1	1	1	0

Booth's algorithm [58] is based on the fact that fewer partial products have to be generated for groups of consecutive '0' in the multiplier there is no need to generate any new partial product. For every '0' bit in the multiplier, the previously accumulated partial product needs only to be shifted by one bit to the right. The above can be implemented by recoding the multiplier as shown in the table 2.1.

Table 2. 2; Multiplier recoding for radix-4 booth's algorithm

Sl. No.	mr_{i+1}	mr_i	mr_{i-1}	Recoded digit	Operation on multiplicand
1	0	0	0	0	0 X Multiplicand
2	0	0	1	+1	+1 X Multiplicand
3	0	1	0	+1	+1 X Multiplicand
4	0	1	1	+2	+2 X Multiplicand
5	1	0	0	-2	-2 X Multiplicand
6	1	0	1	-1	-1 X Multiplicand
7	1	1	0	-1	-1 X Multiplicand
8	1	1	1	0	0 X Multiplicand

It is based on portioning the multiplier in to overlapping group of 3- bits and each group is decoded to generate corresponding partial product. Each recoded digit

performs a certain operation on the multiplicand shown above in the table 2.2. The primary advantage of using this multiplication scheme is that it reduces the number of partial products generated by half the number.

For example consider 6X6 bit multiplication, number of partial products involved will be 3 where as in Add- Shift algorithm six partial products are needed.

Example :						
A		01	00	01	17	Multiplicand
X	x	11	01	11	-9	Multiplier
		-A	+2A	-A		0peration
Add –A +		10	11	11		
2 bit shift		11	10	11	11	
Add 2A		10	00	10		
		01	11	01	11	
2 bit shift		00	01	11	01	11
Add –A +		10	11	11		
		11	01	10	01	11 -153

Figure 2. 20: Add and shift method of partial product generation

n/2=3 steps;2 multiplier bits in each step

All shift operations are 2 bit position shifts

Accumulation of the partial products in multiplication is accelerated by adding all the partial products at a time.

2.5.4 Low Power Multipliers

2.5.4.1 Pipelined Modified Booth Multiplier

A pipelined modified Booth multiplication is proposed to enhance the power performance ratio of 2's complement multiplication. System architecture of the proposed scheme is catered for VLSI implementation. It is designed with the merit of low power consumption achieved by reducing the number of adder required. Only half of the adders is required as in traditional pipelined multiplier [54-56].

Modified Booth algorithm is widely used to implement multiplication in DSP systems and other applications. It provides high performance than other multiplication algorithms. However, the intrinsic architecture of the modified Booth algorithm does not have the regularity for VLSI pipeline implementation. For power reduction, reducing the supply voltage is widely used to improve the system power efficiency. It is most effective in pipelined data path comparing to other implementations such as parallel data path and pipeline-parallel data path. It is desirable to employ pipelining in

multiplication for DSP applications. Elrabaa *et. al.* proposed a sign extension scheme makes the multiplier array more regular for VLSI implementation. Nevertheless, it requires an n-bit adder at the last stage to sum all the terms from the multiplier array that impedes the system throughput of the multiplier. Besides, required 2's complement of multiplicand in Booth algorithm proscribes pipelining.

For reduction of power and supply voltage, pipelined data path is being widely used in DSP systems to enhance the system throughput. In the paper [58], we proposed a pipelined modified Booth multiplication to improve the system performance in terms of system throughput and power-performance ratio for low power low voltage DSP applications.

2.5.4.2 Problems in computing 2's complement number

Multiplying two numbers, *X* (multiplicand) and *Y* (multiplier), Booth algorithm encodes the two's complement multiplier, Y, to reduce the number of partial products to be added. The Radix - 4 modified Booth algorithm [57] divides the multiplier into overlapping groups of 3-bit encoded into {-2,-1,0,1, 2}. Each group is decoded to generate the partial product according the rule shown in Table 1.

The inherent problem in pipelining the modified Booth algorithm is the generation of 2's complement of the multiplicand. In conventional pipelined multiplier, a partial product is generated locally at each stage and added to the partial product from the previous stage with a single addition. However, in modified Booth algorithm, it may require multiple additions at each stage in order to generate the correct partial product that involves 2's complement of the multiplicand (for the codes -1 and -2**).** In these cases, it requires the inversion of the multiplicand and addition of '1' at the least significant bit for the code '-1', or the inversion of the multiplicand and left shift the multiplicand by one bit, and followed by the addition of '10' for the code '-2'. Such addition of '1' or '10' may lead to extra n additions in order to propagate the carry from the least significant bit to the most significant bit of the partial product. In other words, the 2's complement of the multiplicand cannot be finished with one addition and one inversion operation.

2.5.4.3 Modified Booth Pipelined Multiplication

To pipeline the modified Booth multiplication [55-57], a partial product generation scheme is proposed to accommodate the 2's complement "correction" bit/bits required

for the code {-1, -2). Two extra bits are patched at the end of the least significant bit for any partial product. Value of these patched bits are “00”, “01” or “10” depending on the previous partial product encoded with the corresponding set of codes {2,1,0}, {-1}**,** and {-2}.

For 6x6 bits Radix-2 modified Booth multiplication, the original 9-bit partial product is extended to 11 bits: 9-bits partial product with Elrabaa algorithm, and 2-bit 2’s complement “correction”. Figure 2.21 shows the proposed partial product generation. At any stage, it generates a partial product consists of the multiplicand for the codes (2, 1, 0) or its inversion for the codes (-1, -2}, and 2-bit 2’s complement “correction”. The 2’s complement “correction” bits are passed to the next stage. At any stage, the first part of the partial product (the multiplicand or its inversion) and the 2’s complement “correction” bits from the previous partial product are added to the partial product. However, at any intermediate stage except the first stage, it involves an addition of four operands: sum and carry from previous stage (s2, c1), the carry from the second least significant bits from the addition of (s1, c0), and the patched “correction” bit (Yi) indicated as the dashed objects in Figure 2.22**.** For proper addition, the traditional full adder cannot be used in this case.

To solve the problem, an encoding scheme is developed for the patched “correction” bits, XiYi, at any intermediate stage. Values of these patched ”correction” bits are “00”, “01”, or “10” depending on the corresponding set of Booth codes {2,1,0}, {-1-, 2) of the partial product of the previous stage. Since it only involves a 2-bit addition of the second least significant bits (s1,c0), XiYi are recoded to ”0l”, “10” or “11” upon the result of the logic evaluation of “s1 AND c0*’’* that can be achieved by a simple AND gate. In other words, a ‘0’ or ‘1’ is added to X_iY_i depending on the carry from the addition of s1 and c0. The possible result of such operation is “00”, “01”**,”**00**”** or “11”. It never has the case needed to propagate a carry to the more significant bits resulted from this addition as the possible values of X_iY_i are ”00”, “01”,or “10”.

At any intermediate stage, X_iY_i passed from the previous stage is recoded according to the result of the evaluation of the carry from the second least significant bits (s1, c0) before adding to the sums and carries evaluated from the previous stage. In fact, it will not have a large overhead for the recoding scheme to penalize the system performance at any intermediate stages.

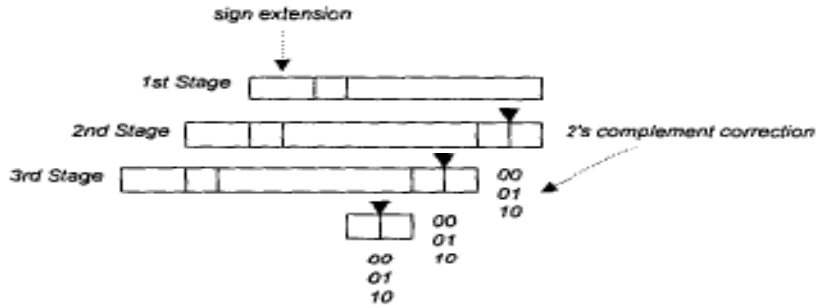


Figure 2. 21: Partial product generation for 6x6 bit modified Booth multiplication

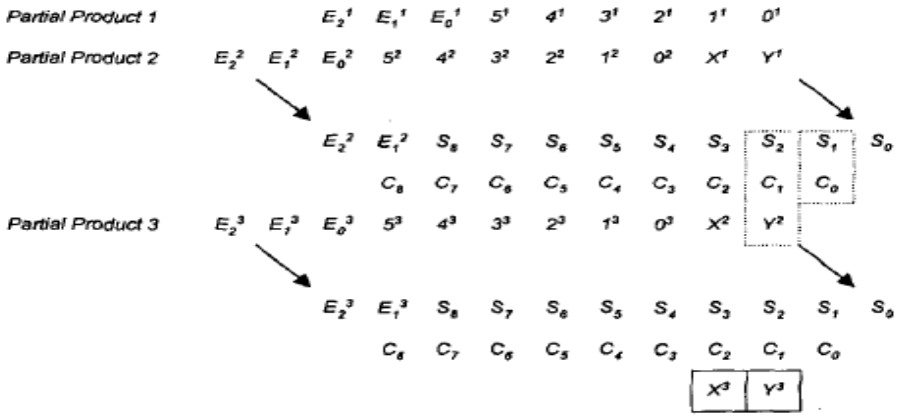


Figure 2. 22: Data flow of a 6 X 6 bit modified Booth multiplication with the partial Product generation scheme

2.5.4.4 Results and Conclusion

The simulation result multipliers using tree structure and pipelined booth multiplier is given bellow.

Power for tree multiplier is 0.667μW Max Pin Delay: 8.113 ns

Power for Pipelined booth multiplier- 0.43μW Max Pin Delay: 7.431 ns

Improved power efficiency of pipelined booth multiplier compared to tree multiplier is about 35.25% and a speed improvement of 8.4% is obtained.

Chapter-III

Decimation Filter for Hearing Aid Application using Distributed Arithmetic

3.1 Introduction

Oversampling analog-to-digital converters are used extensively in audio application as it provides higher performance and flexibility by removing the complexity of the analog circuitry [59-61]. In such applications, over-sampling uses a sampling rate that is much greater than the bandwidth of the signal of interest. Digital signal processing (DSP) techniques are applied to perform further filtering, decimation and even down conversion. The range of human hearing is generally considered to be 20 Hz to 20 kHz, but the ear is far more sensitive to sounds between 1 kHz and 4 kHz [63-64]. Hence it is more useful to design a hearing aid application operating within the specified frequency range. A hearing aid application makes use of the over-sampling concept and consists of a sigma-delta analog-to-digital converter (ADC) followed by a decimation filter [62] as shown in figure 3.1.

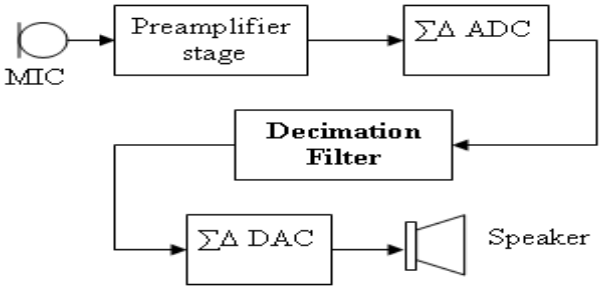


Figure 3. 1: The Decimation filter used for the hearing aid application

For designing this filter, we need a multiplier and adder, and here in this investigation we have used distributed arithmetic for its implementation. Distributed Arithmetic (DA) is so named because the arithmetic operations that appear in signal processing (e.g., addition, multiplication) are not "lumped" in a comfortably familiar fashion, but are distributed in an often unrecognizable fashion. The most-often encountered form of computation in digital signal processing is a sum of products (or in vector analysis parlance, dot-product, or inner-product generation). This is also the computation that is executed most efficiently by DA.

3.2 Distributed Arithmetic

DA is basically (but not necessarily) a bit-serial computational operation that forms an inner (dot) product of a pair of vectors in a single direct step. The advantage of DA [65-66] is its efficiency of mechanization but the disadvantage is its slowness because of bit-serial nature. This disadvantage can be avoided in two ways i.e.

a. It is not real if the time required to input eight 8-bit words one at a time in a parallel fashion is exactly the same as the time required to input (simultaneously on eight wires) all eight words serially.
b. Speed can be increased by employing techniques such as bit pairing or partitioning the input words into the most significant half and least significant half, the least significant half of the most significant half, etc., thereby introducing parallelism in the computation.

As an example of direct DA [65-66] inner-product generation, consider the calculation of the following sum of products as

$$y = \sum_{k=1}^{K} a_k x_k \tag{3. 1}$$

The a_k are fixed coefficients, and the x_k are the input data words. If each x_k is a 2's-complement binary number scaled such that $|x_k|<1$ then each x_k can be presented as

$$x_k = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} \tag{3. 2}$$

where the b_{kn} are the bits, 0 or 1, b_{ko} is the sign bit, and $b_{k,n-1}$ is the least significant bit (LSB).

Now let us combine equations 3.1 and 3.2 to express y in terms of the bits of x_k

$$y = \sum_{k=1}^{K} a_k \left[-b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n} \right] \tag{3. 3}$$

Equation 3.3 is the conventional form of expressing the inner product. Direct mechanization of this equation defines a "lumped" arithmetic computation. Let us interchange the order of the summations, which gives

$$y = \sum_{n=1}^{N-1} \left[\sum_{k=1}^{K} a_k b_{kn} \right] 2^{-n} + \sum_{k=1}^{K} a_k \left(-b_{k0} \right) \tag{3. 4}$$

Here equation 3.4 defines a distributed arithmetic computation. Consider the bracketed term in equation 3.4

$$y = \sum_{k=1}^{K} a_k b_{kn} \tag{3. 5}$$

Because each b_{kn} may take on values of 0 and 1 only, expression (3.5) will have only 2^K possible values. Rather than computing these values on line, we can precompute these values and store them in a ROM. The input data can be used to directly

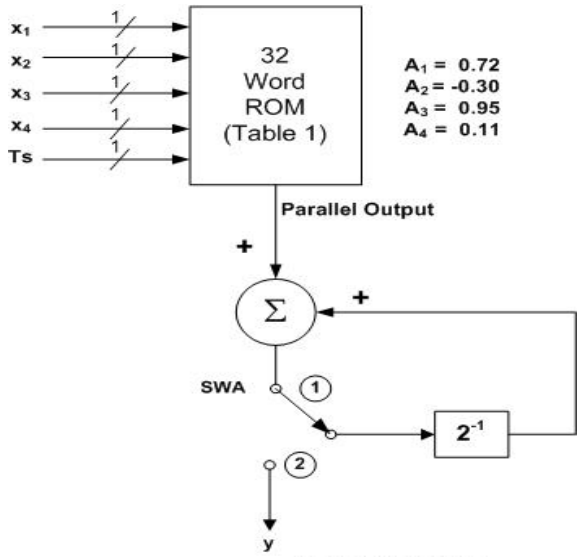
address the memory and the result, i.e. the $\sum_{k=1}^{K} a_k b_{kn}$, can be put into an accumulator. After *N* such cycles, the memory contains the result, *y*.

As an example, let K = 4, A_1, = 0.72, A_2, = -0.30, A_3 = 0.95, and A_4 = 0.11. The memory must contain all possible combinations (2^4 = 16 values) and their negatives in order to accommodate the term

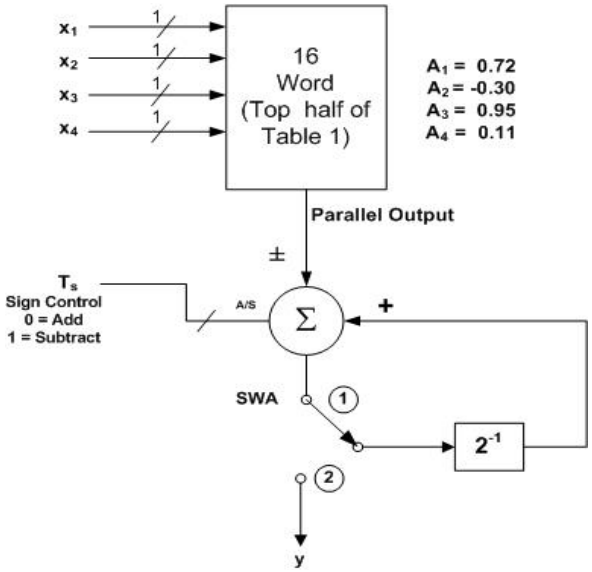
$$\sum_{k=1}^{K} a_k \left(-b_{k0}\right) \tag{3. 6}$$

which occurs at the sign-bit time. As a consequence, we need to use a 2×2^K word ROM. Figure 3.2(a) shows the simple structure (with a 2×2^4 = 32-word ROM) that can be used to mechanize these equations.

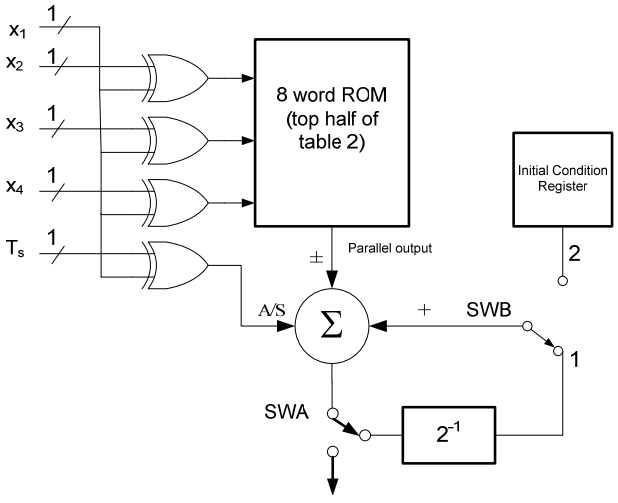
Table 1 shows the contents of the memory. The T_s, signal is the sign-bit timing signal. We assume that the data on the x_1, x_2, x_3, and x_4 lines (which with *T,* comprise the ROM address words) are serial, 2's-complement numbers. Each is delivered in a one-bit-at-a-time (1BAAT) fashion, with LSBs {b_{kn}, n-1} first. The sign bits {b_{k0}} are the last bits to arrive. The clock period in which the sign bits all simultaneously arrive is called the "sign-bit time". During the sign-bit time the control signal T_s = 1, otherwise T_s = 0. For the moment we will assume essentially zero time delay between the time of arrival of the address pattern to the ROM and the availability of its output. The delay around the accumulator loop is assumed to be one clock cycle and is concentrated in the summer. Switch SWA remains in Position 1 except during the clock cycle that follows the sign-bit time, when it toggles for one clock cycle to position 2, and the fully formed result is output.



(a) adder and Full Memory



(b) Adder/ Subtractor and Memory



(c) Adder/ Subtractor and Reduced Memory

Figure 3. 2 Structure of ROM for K=4

Table 3. 1: ROM content for 32 words with K=4

	Input code					32-word
	T_s	b_{1n}	b_{2n}	b_{3n}	B_{4n}	memory content
$1 \leq n \leq N-1$	0	0	0	0	0	0
	0	0	0	0	1	$a_4 = 0.11$
	0	0	0	1	0	$a_3 = 0.95$
	0	0	0	1	1	$a_3+a_4 = 1.06$
	0	0	1	0	0	$a_2 = -0.30$
	0	0	1	0	1	$a_2+a_4 = -0.19$
	0	0	1	1	0	$a_2+a_3 = 0.65$
	0	0	1	1	1	$a_2+a_3+a_4 = 0.75$
	0	1	0	0	0	$a_1 = 0.72$
	0	1	0	0	1	$a_1+a_4 = 0.83$
	0	1	0	1	0	$a_1+a_3 = 1.67$
	0	1	0	1	1	$a_l+a_3+a_4 = 1.78$
	0	1	1	0	0	$a_1+a_2 = 0.42$
	0	1	1	0	1	$a_1+a_2+a_4 = 0.53$
	0	1	1	1	0	$a_1+a_2+a_3 = 1.37$
	0	1	1	1	1	$a_1+a_2+a_3+a_4 = 1.48$
$n=0$	0	0	0	0	0	0
	0	0	0	0	1	$-a_4 = -0.11$
	0	0	0	1	0	$-a_3 = -0.95$
	0	0	0	1	1	$-(a_3+a_4) = 1.06$
	0	0	1	0	0	$-a_2 = -0.30$
	0	0	1	0	1	$-(a_2+a_4) = -0.19$
	0	0	1	1	0	$a_2+a_3 = 0.65$
	0	0	1	1	1	$a_2+a_3+a_4 = 0.75$
	0	1	0	0	0	$a_1 = 0.72$
	0	1	0	0	1	$a_1+a_4 = 0.83$
	0	1	0	1	0	$a_1+a_3 = 1.67$
	0	1	0	1	1	$a_l+a_3+a_4 = 1.78$
	0	1	1	0	0	$a_1+a_2 = 0.42$
	0	1	1	0	1	$a_1+a_2+a_4 = 0.53$
	0	1	1	1	0	$a_1+a_2+a_3 = 1.37$
	0	1	1	1	1	$a_1+a_2+a_3+a_4 = 1.48$

We may reduce the memory size by half to a 2^K word ROM by modifying the adder to an adder/ subtractor and using T_s as the add-subtract-control line as shown in figure 3.2(b). This configuration may now be mechanized with a 16-word ROM. The stored table is simply the upper half of table 3.1.

The memory size may be halved again to ½2^K words. In order to understand how this works, we shall interpret (not convert, but just interpret) the input data as being cast not in a (0, 1) straight binary code, but instead as being cast in a (-1, 1) offset binary code. Suppose that we think of x_k as

$$x_k = \frac{1}{2}\left[x_k - (-x_k)\right] \tag{3. 7}$$

and also in 2's-complement notation the negative of x_k is written as

$$-x_k = -\bar{b}_{k0} + \sum \bar{b}_{kn}\, 2^{-n} + 2^{-(N-1)} \tag{3. 8}$$

where the over score symbol indicates the complement of a bit. From equations 3.2 and 3.8 we may rewrite the equation 3.7 as

$$x_k = \frac{1}{2}\left[-(b_{k0} - \bar{b}_{k0}) + \sum_{n=1}^{N-1}(b_{kn} - \bar{b}_{kn})2^{-n} - 2^{-(N-1)}\right] \tag{3. 9}$$

In order to simplify our notation later, it is convenient to define the new variables

$$c_{kn} = b_{kn} - \bar{b}_{kn}, \qquad n \neq 0 \tag{3. 10}$$

and

$$c_{k0} = \left(b_{k0} - \bar{b}_{k0}\right) \tag{3. 11}$$

where the possible values of the $\boldsymbol{c_{kn}}$, including $n = 0$, are C1. Now equation 3.9 may be rewritten as

$$x_k = \frac{1}{2}\left[\sum_{n=0}^{N-1} c_{kn}\, 2^{-n} - 2^{-(N-1)}\right] \tag{3. 12}$$

By substituting (3.12) into (3.1) we obtain

$$y = \frac{1}{2}\sum_{k=1}^{K} a_k \left[\sum_{n=0}^{N-1} c_{kn}\, 2^{-n} - 2^{-(N-1)}\right] \tag{3. 13}$$

$$y = \frac{1}{2}\sum_{n=0}^{N-1} Q(b_n)2^{-n} + 2^{-(N-1)}Q(0) \tag{3. 14}$$

where

$$Q(b_n) = \sum_{k=1}^{K} \frac{a_k}{2} c_{kn} \quad and \quad Q(0) = \sum_{k=1}^{K} \frac{a_k}{2} \tag{3. 15}$$

The $Q(b_n)$ has only $2^{(k-1)}$ possible amplitude Values with a sign that is given by the instantaneous combination of bits. This is consistent with our earlier claim. The computation of *y* is mechanized using a 2^{K-1} word memory, a one-word initial condition register for Q(0) , and a single parallel adder/ subtractor with the necessary control-logic gates. This is shown in figure 3.2c using the 8-word ROM, which contains the $Q(b_0)$.

Table 3. 2: Reduced ROM content for 32 word

Input code				32-word
b_{1n}	b_{2n}	b_{3n}	b_{4n}	memory content
0	0	0	0	$-1/2\ (a_1+a_2+a_3+a_4) = -0.74$
0	0	0	1	$-1/2\ (a_1+a_2+a_3-a_4) = -0.63$
0	0	1	0	$-1/2\ (a_1+a_2-a_3+a_4) = 0.21$
0	0	1	1	$-1/2\ (a_1+a_2-a_3-a_4) = 0.32$
0	1	0	0	$-1/2\ (a_1-a_2+a_3+a_4) = -1.04$
0	1	0	1	$-1/2\ (a_1-a_2+a_3-a_4) = -0.93$
0	1	1	0	$-1/2\ (a_1-a_2-a_3+a_4) = -0.09$
0	1	1	1	$-1/2\ (a_1-a_2-a_3-a_4) = 0.02$
1	0	0	0	$½\ (a_1-a_2-a_3-a_4) = -0.02$
1	0	0	1	$1/2\ (a_1-a_2-a_3+a_4) = 0.09$
1	0	1	0	$1/2\ (a_1-a_2+a_3-a_4) = 0.93$
1	0	1	1	$1/2\ (a_1-a_2+a_3+a_4) = 1.04$
1	1	0	0	$1/2\ (a_1+a_2-a_3-a_4) = -0.32$
1	1	0	1	$1/2\ (a_1+a_2-a_3+a_4) = -0.21$
1	1	1	0	$1/2\ (a_1+a_2+a_3-a_4) = 0.63$
1	1	1	1	$1/2\ (a_1+a_2+a_3+a_4) = 0.74$

It may noticed from the memory values of table 3.2 that those values in the lower half under "Q" are the mirror images of the values in the upper half, but with the signs reversed. If we look at the bit patterns in the left-hand column, we find that if we XOR b_{1n}, with the remaining set of b_{2n}, b_{3n} and b_{4n} we properly address the 8-

word memory to pull out the correct values of Q except for the sign. By using the b_{1n} as the add/subtract control line for the accumulator input, we also now have the proper sign. During the sign-bit time the add/subtract command must be inverted. We therefore combine the b_{ln}, and T_s signals through an XOR gate in order to derive the proper add/subtract control signal.

The initial-condition memory that contains the value Q(0) is shown on the extreme right side of figure 3.2(c). When the LSBs of the x_k are addressing the ROM, the value that is read out from the ROM must be corrected by the Q (0) through switch SWB, which operates synchronously with switch SWA. This artefact of the binary-offset system can be seen in equation 3.14. Subsequent values from the ROM are summed with the shifted previous sum. As before, we assume zero time delay between the application of the addressing bits and the availability of the contents of the ROM. There is a delay of one clock through the parallel adder, and the switches SWA and SWB are in position *2* only for the clock cycle following the sign-bit time when $T_s = 1$. During the first clock cycle, the first output from the ROM, $Q(b_{N-1})$,is summed with Q(0)*;* during the second clock cycle it is right shifted and summed with $Q(b_{N-2})$ to produce $Q(b_{N-2})+ [Q(b_{N-1})+ Q(O)]2^{-1}$ during the third clock cycle it is again right shifted and summed with $Q(b_{N-3})$to produce $Q(b_{N-2})+ [Q(b_{N-2})2^{-1}+ Q(b_{N-2})+ [Q(b_{N-1})+ Q(O)]2^{-1}$ up to the Nth clock cycle when we add $Q(b_0)$ to the right shifted previously computed quantity to produce $Q(b_0) +Q(b_1)2^{-1}+ Q(b_2)2^{-2}+ \cdots + Q(b_{N-2})2^{-(N-2)}+ [Q(b_{N-1})+ Q(O)]2^{-(N-1)}$.

3.3 Oversampling Technique for Analog to Digital Conversion

As the cost of VLSI chips decreases there is a greater demand on the analog-digital interfaces between these VLSI circuits and the analog world. Due to the large number of analog-digital interfaces in the VLSI systems, low-cost monolithic MOS analog-digital conversion techniques have been an important element in reducing the cost of the total systems. Also the MOS technology sizes continues to fall which gives rises to high speed, programmability and low production cost makes digital signal processors a more attractive alternative to conventional analog signal processing. To exploit the significant computational power, a high resolution analog to digital interfaces using oversampled concept with particular emphasis on sigma-delta structures [57-62] is considered and discussed in next section.

3.3. 1 Comparison Between Oversampled and Nyquist Rate A/D Converters

The primary functions of an A/D converter that converts a continuous time analog input signal x(kT) into a sequence of digital code y(kT*)* at a rate f_N is shown in figure 3.3. To prevent aliasing, the analog filter in front of the sampling block in figure 1 limits the bandwidth of x(kT) to half the sampling rate. The sampled signal is then converted to digital codes y(kT) by the quantizer.

Figure 3. 3: AD converter block diagram

A/D converter architectures [57-59] will now be subdivided into two groups based on the sampling rate of the converter as followings:

a. Nyquist Sampling A/D Converter.
b. Oversampling A/D Converter.

The Nyquist rate converters are defined to be converters that sample the analog signal at its Nyquist frequency $f_N = 2W$, where W is the passband of the converter input. Oversampled converters are used to denote converters that sample at a much higher rate $f_s >> f_N$. The ratio of the sampling frequency to the Nyquist rate is defined as the oversampling ratio, D. In many applications the bandwidth of the signal is small compared to the typical speed of operation available in VLSI circuits and use of oversampled A/D converters offers improved resolution while achieving same performance.

3.3.1.1 Anti-Aliasing

Nyquist rate converters and oversampled converters differ in two fundamental aspects like anti-alias filtering and quantization. In the Nyquist rate converters, the filtering is done before the A/D converter using complex analog filters. In the case of oversampled converters the filtering can be done in multiple stages. The analog anti-aliasing filter preceding the A/D converter has a relaxed requirement as there exists large number of undesired states.. The latter stages of filtering for subsequent decimation can be implemented digitally, taking advantage of digital signal processing capability provided by the VLSI technology.

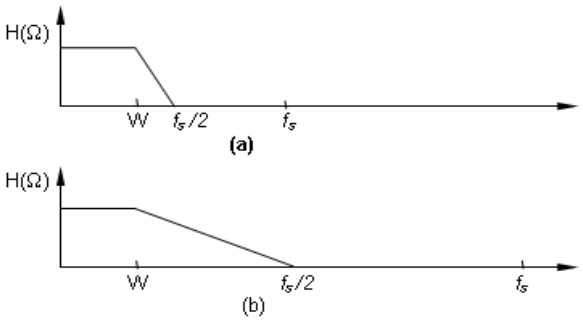


Figure 3. 4: Requirements of anti-aliasing filter (a) Nyquist rate ADC (b) Oversampling ADC

From figure 3.4 it can be seen that Nyquist rate converter requires an antialiasing filter with very sharp transition in order to provide adequate aliasing protection but the transition band of anti-aliasing filter of oversampled A/D converter have much wider than its passband. Since anti-aliasing suppression is required only for a small band of frequencies around half the sampling rate, $f_s/2$, this simplifies the design of antialiasing filter.

3.3.1.2 Quantization

If the analog samples can be represented exactly, there is no apparent advantage to sampling at a rate higher than the Nyquist rate. But during the analog to digital conversion, quantization has to be performed, resulting in quantization errors. If more samples are taken, these errors can be reduced. For example, if the number of samples taken per unit time is doubled and each pair of samples produced is averaged to produce new samples at the original sample rate, the signal component of the samples will add linearly while the quantization noise component in the samples will add as random variables. If the noise is uncorrelated then the signal to noise ratio will improve by 3 dB. On the other hand, for a Nyquist rate A/D converter quantization is performed for every sampling instant at the full precision of the converter.

3.3.2 Operation of a Simple Sigma-Delta Modulator

A simple oversampled modulator, called the first-order sigma-delta modulator, is shown in figure 3.5. It consists of a summing node, an integrator, and one-bit A/D and D/A (Digital to Analog) converters in a feedback loop. Since the integrator has infinite gain at dc, the loop gain is infinite at dc and therefore the dc component or the average of the error signal is zero. Consequently, the dc component or the average of the output from the DAC (Digital to Analog Converter) [60-64] will be identical to the dc component of the input signal. This means that even though the quantization

error at every sample is large because a quantizer with only two levels is being used, the average of the quantized signal, and therefore the modulator output *y(kT),* tracks the analog input signal *x(t).* This average is computed by the decimation filter that follows the converter. In general, the integrator output ramps up and down according to the value of the DAC. Correspondingly the 1-bit ADC outputs a bit stream of ones and zeros which is a pulse density modulated representation of the input dc value.

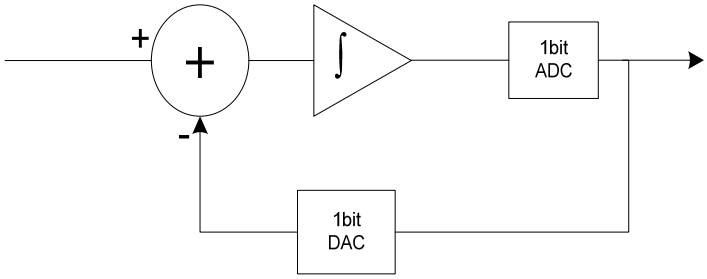


Figure 3. 5: Oversampled delta modulator

In general the amplitude resolution of the converter increases when more samples are included in the averaging process, or as the oversampling ratio D increases. However, the bandwidth is decreased at the same time. Consequently, the resolution of oversampled A/D converters is a function of ratio of the sampling rate to the bandwidth of the converter. The principle of operation for the oversampled A/D converters relies on this fundamental trade-off between resolution and time.

3.3.3 Modulators for Oversampled A/D Conversion

3.3.3.1 Classification

The quantizer in an A/D [59-64] converter performs quantization by approximating the analog input with the nearest lower level from a set of discrete reference levels. The quantization error is defined as the difference between the quantizer input and the value of the output. The variance of the quantization noise is a measure of the quantizer resolution and in a Nyquist rate converter is reduced by increasing the number of reference levels. In comparison, the oversampled A/D [60-65] converter reduces the quantization noise within the signal baseband, for a quantizer with a given resolution, through the use of feedback.

3.3.3.2 Predictive Coders

The block diagram in figure 3.5 illustrates the basic architecture of a predictive or differential modulator. For a first-order predictive coder, the transfer function

H(z) will be an integrator. It works on the principle that if the course of a signal up to a certain point in time is known, it is possible to make some inference about its continuation. This extrapolation process is called prediction. To achieve predictive encoding, the value of the present sample is used to calculate the next sample. The difference between the actual value and the estimation is quantized by the 1-bit quantizer. The quantized value is then stored in the integrator. Since the error is encoded in digital form, to calculate the analog predicted value, a DAC is required in conjunction with either an analog or digital integrator. As an analog to digital interface the advantage offered by prediction is that it reduces the dynamic range of the ADC by subtracting an estimate $\hat{X}(t)$ from the modulator input *x(t).* The step size of the quantizer is adjusted to this reduced range, which results in a decrease of the quantization noise.

3.3.3.3 Noise Shaping Coders

Noise shaping modulators do not reduce the magnitude of the quantization noise, but instead shape the power density spectrum of this error, moving the energy towards high frequencies. Provided that the analog input to the quantizer is oversampled, the high frequency quantization noise can be eliminated with a digital lowpass filter without affecting the signal band. figure 3.5 shows the noise shaping function of the first order coder presented in figure 3.6.

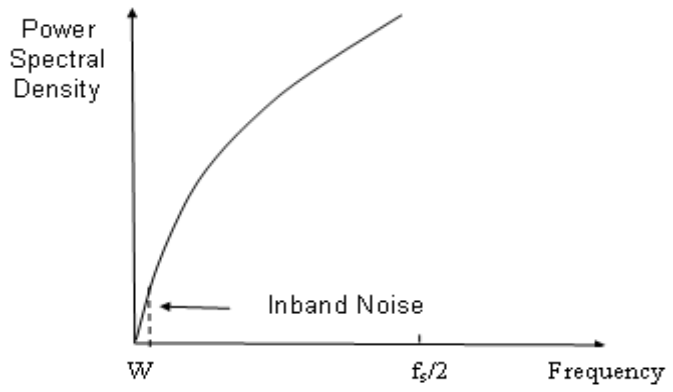


Figure 3. 6: Power Spectral density for 1st order sigma-delta coder

3.3.3.4 Comparison

Commonly known delta modulators can be used as predictive and noise shaping modulators and, perform an implicit differentiation of the input signal. If there is a mismatch between the integrators in the encoder and decoder, the signal can be dis-

torted and delta modulators exhibit slope overload for rapidly rising input signals and their performance dependents on the frequency of the input signal. Another type of converters denoted as interpolative modulators and these converters combine both the prediction and noise shaping techniques. A muitibit DAC is needed in the feedback loop that must be accurate because it determines the overall precision of the converter. As in the case of predictive coding, the performance of interpolative modulators is susceptible to the slope of the input signal.

Noise shaping modulators encode the signal itself and their performance is insensitive to the rate of change of the signal. When the internal ADC and DAC is implemented with a one bit resolution, the modulator is called delta-sigma modulator or sigma-delta modulator. A second- order oversampled modulator shown in figure 3.6 has been selected as an example. Three configurations are shown: a 2nd-order noise shaping, a lst-order predictive with lst-order noise shaping or a 2nd-order predictive converter as shown in figure 3.7. Signal transfer function (STF) and noise transfer functions (NTF) for each structure are shown in table 3.3.

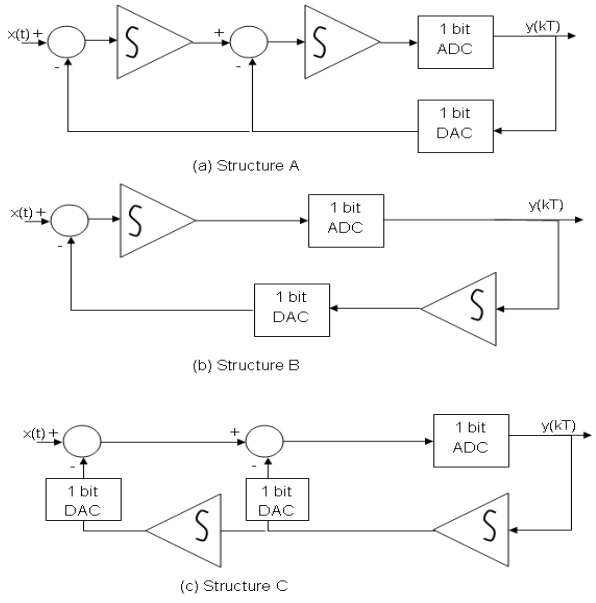


Figure 3. 7: Three different structures of ∑Δ modulators

Table 3. 3: Transfer function of three structures

Structure	STF	NTF
A	1	$(1-z^{-1})^2$
B	$(1-z^{-1})^{-1}$	$(1-z^{-1})$
C	$(1-z^{-1})^{-2}$	1

Each structure has no major difference in terms of performance, but significant differences arise in actual VLSI implementation. First of all, integrators in A must be realized with analog circuits, the first integrator in B and both integrators in C can be realized digitally followed by a DAC. With this arrangement, the number of operational amplifiers required will be 2, 1, and 0 for A, B, and C, respectively. The resolution or number of bits required for the DAC is different. As an example, if we assume the oversampling ratio is 128, then the resolution needed for the DAC will be 1, 6, and 12 for A, B, and C, respectively. In cases B and C, non-idealities introduced by the integrators and will limit the linearity of the modulator. For the decimation filter [62-63], multipliers operating at the oversampling frequency f_s are required for the cases in B and C. With a second-order high pass quantization noise spectrum the highest out-of-band attenuation is required for the decimation filter in A. The differences between the three cases are now summarized in table 3.4.

Table 3. 4: Comparison among three structures

Structure	A	B	C
Number of analog stage	2	1	0
DAC resolution	1	medium	high
Complexity of decimation filter	large	medium	small
High-speed multiplier	no	yes	yes
Decimation filter attenuation	high	medium	low

3.4 Half-band filter

Linear-phase FIR half-band filters have found several applications and for instance, in the design of sharp cutoff FIR filters, a multistage design based on half-band filters [65-66] is very efficient. Half-band FIR digital filters are known to be important due to their reduced computational requirements as about 50 percent of the filter coefficients are zero, thus, cutting down the implementation cost.. They are especially valuable when used to decimate or interpolate by a factor of 2, where their characteristics well match the filtering requirements. Computational savings can also be achieved by extending the concept of half-band filters to Nth-band filters for use in decimation or interpolation by a factor of N.

Let H(z) denotes the transfer function of a (linear-phase, FIR) half-band filter of order N – 1, which is given by equation 3.16.

$$H(z)=\sum_{n=0}^{N-1}h(n)z^{-n}\text{ , h(n) is real} \qquad (3.16)$$

These filters are restricted to be of Type 1 (i.e., N - 1 is even and h(n) = h(N - 1 - n)). The frequency response is thus of the form $H(e^{j\omega}) = e^{-j\omega(N-1)/2} H_0(e^{j\omega})$, where $H_0(e^{j\omega})$ represents the real-valued amplitude response. A typical plot of $H_0(e^{j\omega})$ is shown in figure 3.8, assuming an equiripple type of design. There is symmetry with respect to the half-band frequency $\pi/2$, i.e., the band edges are related as

$$\omega_p + \omega_s = \pi \qquad (3.17a)$$

And the ripples are related as

$$\delta_1=\delta_2=\delta \qquad (3.17b)$$

In view of symmetry the impulse response h(n) satisfies

$$h(n)=\begin{cases}0, & n-\dfrac{N-1}{2}=even\ and\ nonzero\\ \dfrac{1}{2}, & n=\dfrac{N-1}{2}\end{cases} \qquad (3.18)$$

The simplest way to design equiripple half-band filters is to invoke the widely used McClellan-Parks algorithm with the specifications satisfying (3.17a) and (3.17b). (If equiripple nature is not a requirement, then window designs are the fastest) The resulting filter satisfies (3.18) with reasonable accuracy.

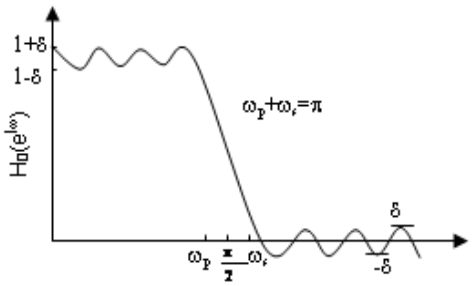


Figure 3. 8: Amplitude response of a half-band FIR filter

The only disadvantage with this procedure is that those coefficients which are supposed to satisfy are treated as unknowns in the optimization, and, accordingly, takes more time to design.

The design time can be reduced considerably by exploiting the partial knowledge about the impulse response coefficients. The technique also leads to a structural

interpretation of half-band filters, which enables us to implement these filters in such a way that, if the structure has low pass-band sensitivity, then it automatically has low stop-band sensitivity as well. (This is significant in view of the fact that low pass-band and low stop-band sensitivities are often conflicting requirements.) We conclude by indicating how the ideas can be extended for M^{th} band filters (which find application in decimation and interpolation filters). Design examples are presented demonstrating the significant features of the results.

In view of equation (3.18), we can assume (N - 1)/2 to be odd. (Indeed, if (N - 1)/2 were even, then (3.18) would imply h(0) = h(N - 1) = 0; by redefining h(1) to be h(0), we can cut down the order to N - 3.) Given the specifications ω_p, ω_s, and π, let us first design a one-band prototype linear-phase filter G(z) of order (N - 1)/2 with specifications as shown in figure 3.8. G(z) has a zero at o = rr, since (N - 1)/2 is odd. Its passband extends from 0 to $2\omega_p$ and the transition band is from $2\omega_p$, to π. If we now define

$$H(z)=\frac{G(z^2)+z^{-N-1/2}}{2} \tag{3. 19}$$

then H(z) is a half-band filter, with specifications as in figure 3.9. The conditions (3.17a), (3.17b), and (3.18) are satisfied exactly. The impulse response of H(z) is evidently related to that of G(z) by

$$h(n)=\begin{cases}\frac{1}{2}g\left(\frac{n}{2}\right), & \text{n even}\\ 0, & \text{n odd } \neq \frac{N-1}{2}\\ \frac{1}{2}, & \text{n}=\frac{N-1}{2}\end{cases} \tag{3. 20}$$

G(z) can be designed with the help of the McClellan-Parks program. This design time is considerably lower than the time required to design H(z) directly, since the order of G(z) is only (N - 1)/2. Moreover, for large N - 1, the design-accuracy is better.

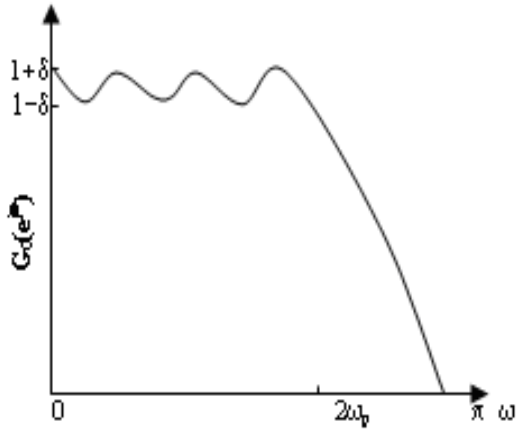


Figure 3. 9: Amplitude response

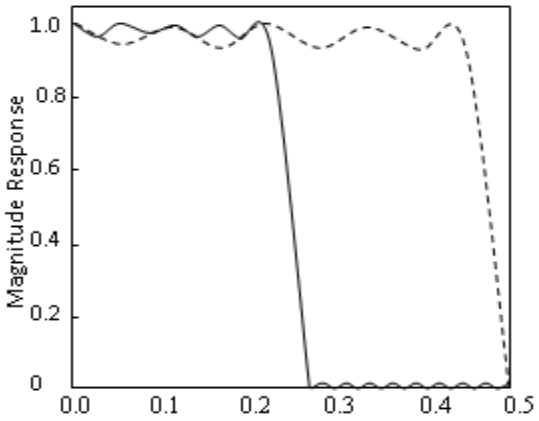


Figure 3. 10: Normalized frequency

A half-band linear phase FIR filter of order N - 1 = 34, and $\omega_p = 0.4577$ is designed using the above method. The magnitude responses of G($e^{j\omega}$) and H($e^{j\omega}$)are shown in figure 3.10.

3.4.1 Low Sensitivity Structures for Half-band Filters

It is well known that a digital filter structure having low pass-band sensitivity does not necessarily have low stop-band sensitivity, and vice versa [67-68]. The coefficient-sensitivity problem in FIR structures has been analyzed in the past. Based on the notion of structural passivity, certain lattice structures can be used to synthesize low-sensitivity structures for any arbitrary FIR transfer function.

The lattice structures satisfy two crucial properties: first, they provide very low pass-band sensitivity; second, if the transfer function has linear phase, this linearity is maintained even when the lattice coefficients are quantized. Now assume that we first implement the one-band filter G(z) using such a structure. Then G(z) has how

pass-band sensitivity. When the lattice coefficients are quantized, the magnitude response of the transfer function G,(z) remains very close to G(z). Since G,(z) retains linear phase and has odd order (N - 1)/2, it continues to have the zero at w = n despite quantization. Suppose we realize H(z) from this structure for G(z) see figure 3.11. Then the stop-band response of H(z) is exactly an image of its pass-band response, even if the coefficients of the lattice are quantized. Thus, H,(z) (the response of the quantized lattice) continues to remain a half-band filter and has low pass-band as well as stop-band sensitivities.

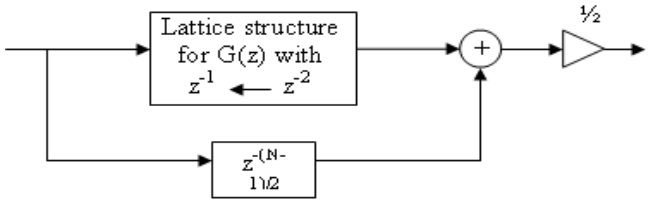


Figure 3. 11: Structure of half-band filter

3.5 Decimation

Fundamental concept that remains behind the digital signal processing is the sampling which provides a set of numbers which is an approximation of the original signal. If a continuous signal presented by $x_C(t)$, $-\infty \leq t \leq \infty$ where x_C is a continuous function of the continuous variable *t (t* may represent time, space, or any other continuous physical variable), then the set of samples can be presented as $x_D(n)$, $-\infty \leq n \leq \infty$ where the correspondence between *t* and *n* is essentially specified by the sampling process, i.e.,

$$n=q(t) \tag{3.21a}$$

Many sampling methods can be used including non-uniform sampling, uniform sampling, and multiple function uniform sampling etc [63]. The most common form of sampling and the one which is used is the uniform (periodic) sampling in which

$$q(t) = \frac{t}{T} = n \tag{3.21b}$$

i.e. the samples $x_D(n)$ are uniformly spaced in the dimension t, occurring nT apart. For uniform sampling we define the Sampling period as T and the sampling rate as

$$F = \frac{1}{T} \tag{3.22}$$

From above equation there exists a unique relation between $x_C(t)$ and $x_D(n)$, if the sampling period T be chosen to satisfy the requirements of the Nyquist sampling theorem and it determines the convenience, efficiency, and / or accuracy in which the signal processing can be performed. In some cases an input signal may already be sampled at some predetermined sampling period T and later it can be convert into sampled signal with new sample period T' such that the resulting signal corresponds to the same analog function.

The process of digitally converting the sampling rate of a signal from a given rate F = 1/T to a different rate *F'* = 1/T is called sampling rate conversion. When the new sampling rate is higher than the original sampling rate the process is generally called interpolation and the reverse is called decimation. Let us take for our design the decimation in signal processing.

3.5.1 Basic Concepts of Sampling rate Conversion

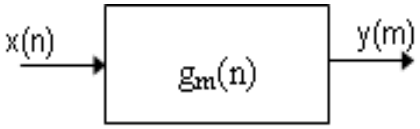


Figure 3. 12: Basic process of sampling rate conversion

Figure 3.12 provides a general description of a sampling rate conversion system. We are given the signal x(n), sampled at the rate F = 1/T, and wish to compute the signal *y* (m) with a new sampling rate F'= *l/T'* assuming that the ratio of sampling periods of y (m) and x (n) *can* be expressed as a rational fraction, i.e.

$$\frac{T'}{T} = \frac{F}{F'} = \frac{M}{L} \tag{3.23}$$

where M and L are integers.

By visualizing the figure 3.12 sampling rate conversion are inherently linear time-varying systems, i.e., $g_m(n)$ is the response of the system at the output sample time m to an input at the input sample time [mM/L]-*n* **.** Since the system is linear, each output sample y(m) can be expressed as

$$y(m) = \sum_{n=-\infty}^{\infty} g_m(n) x\left(\left[\frac{mM}{L}\right] - n\right) \tag{3.24}$$

From equation 3.24 it can be seen that the system response $g_m(n)$ is periodic in m with period *L* ,i.e.,

$$g_m(n) = g_{m+rL}(n),\ r = 0, \pm1, \pm2, \ldots \tag{3.25}$$

In the trivial case when T' = *T,* or L = *M* = 1, equation 3.25 reduces to simple digital convolution the time-invariant equation

$$y(m) = \sum_{n=-\infty}^{\infty} g_m(n)x(m-n) \tag{3.26}$$

Since the period of $g_m(n)$ in this case is 1, and the integer part of m- n is the same as m-n.

3.5.2 Sampling Rate Reduction-Decimation by an Integer Factor M

Consider the process of reducing the sampling rate of x(n) by an integer factor M, i.e.,

$$\frac{T'}{T} = \frac{M}{1} \tag{3.27}$$

Then the new sampling rate is F'= F/M. Assume that x(n) represents a full band signal, i.e., its spectrum is nonzero for all frequencies fin the range -F/2 ≤ f ≤ F/2, with $\omega = 2\pi fT$, i.e.,

$$\left|X\left(e^{j\omega}\right)\right| \neq 0, \quad |\omega| = 2\pi fT \leq \frac{2\pi FT}{2} = \pi \tag{3.28}$$

except possibly at an isolated set of points. Based on sampling theory, in order to lower the sampling rate and to avoid aliasing at this lower rate, it is necessary to filter the signal x (n) with a digital low-pass filter which approximates the ideal characteristic

$$\tilde{H}\left(e^{j\omega}\right) = \begin{cases} 1, & |\omega| \leq 2\pi F'T/2 = \pi/M \\ 0, & otherwise \end{cases} \tag{3.29}$$

The sampling rate reduction is then achieved by forming the sequence *y(m)* by extracting every Mth sample of the filtered output. This process is illustrated in figure 3.13. If we denote the actual low-pass filter unit sample response as k(n), then we have

$$w(n) = \sum_{n=-\infty}^{\infty} h(k)x(n-k) \tag{3.30}$$

where *w(n)* is the filtered output as seen in figure 3.13.

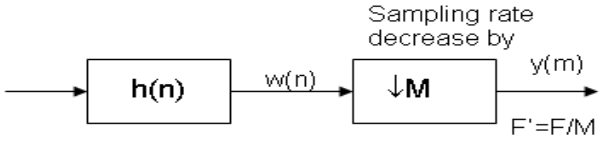


Figure 3. 13: Sampling rate reduction by factor M

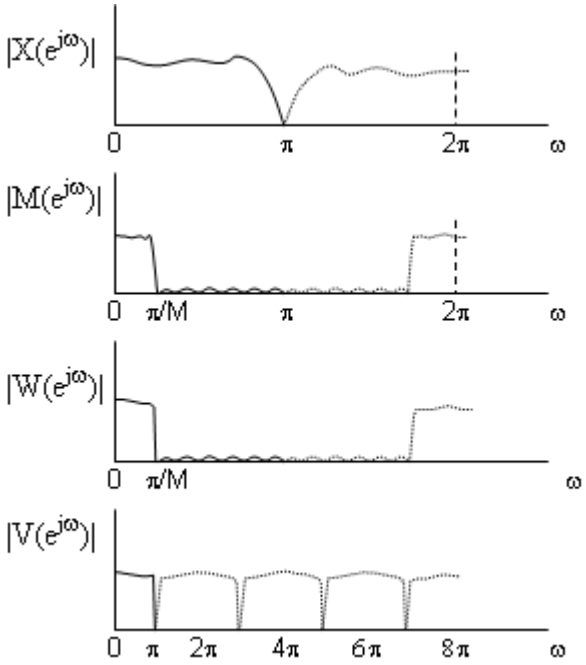


Figure 3. 14: Block diagram typical spectra for sampling rate reduction by factor M

The final output y (m) is then obtained as

$$y(m)=w(Mm) \tag{3.31}$$

as denoted by the operation of the second box in figure 3.13. This block diagram symbol, which will be referred to as a sampling rare compressor given by equation 3.31, figure 3.14 shows typical spectra (magnitude of the Fourier transforms) of the signals x(n), h(n), w(n), and y(m) for an M to 1 reduction in sampling rate.

By combining (3.30) and (3.31) the relation between y (m) and x (n) is of the form

$$y(m) = \sum_{k=-\infty}^{\infty} h(k)x(Mm - k) \tag{3.32}$$

which is seen to be a special case of (3.26). Thus for decimation by integer factors of M we have

$$g_m(n)= g(n)= h (n), \text{ for all m and all n.} \tag{3.33}$$

Though $g_m(n)$ is not a function of m for this case, it can be seen that the system of (3.33) and figure 3.14(a) is not time-invariant by considering the output signal when x(n) is shifted by an integer number of samples. For this case, unless the shift is a multiple of M, the output is not a shifted version of the output for 0 shift, i.e.

$$x(n) \rightarrow y(m) \tag{3.34a}$$

but

$$x(n-\delta) \rightarrow y(m-\delta/M) \text{ unless } \delta = rM \tag{3.34b}$$

To study the nature of the errors in *y(m)*caused by the imperfect low-pass filter let us define a signal

$$w'(n) = \begin{cases} wn) & n = 0, \pm M, \pm 2M, \ldots \\ 0, & otherwise \end{cases} \tag{3.35}$$

i.e., w'(n) = w(n) at the sampling instants of y (m) , but is zero otherwise.
A convenient and useful representation of w'(n) is then

$$w'(n) = w(n)\left\{ \frac{1}{M} \sum_{l=0}^{M-1} e^{j 2\pi \ln / M} \right\} \quad -\infty < n < \infty \tag{3.36}$$

where the term in brackets corresponds to a discrete Fourier series representation of a periodic impulse train with a period of M samples. Thus we have

$$y(m)=w'(Mm)=w(Mm) \tag{3.37}$$

We now write the z-transform of y(m) as

$$Y(z) = \sum_{m=-\infty}^{\infty} y(m) z^{-m} = \sum_{m=-\infty}^{\infty} w'(Mm) z^{-m} \tag{3.39}$$

and since *w'(m)* is zero except at integer multiples of M, equation (3.39) becomes

$$\begin{aligned} Y(z) &= \sum_{m=-\infty}^{\infty} w'(m) z^{-m/M} = \sum_{m=-\infty}^{\infty} w(m) \left[\frac{1}{M} \sum_{l=0}^{M-1} e^{j 2\pi lm / M} \right] z^{-m/M} \\ &= \frac{1}{M} \sum_{l=0}^{M-1} \left[\sum_{l=0}^{M-1} w(m) e^{j 2\pi lm / M} z^{-m/M} \right] = \frac{1}{M} \sum_{l=0}^{M-1} w(m) \left(e^{-j 2\pi lm / M} z^{1/M} \right) \end{aligned} \tag{3.40}$$

Since

$$W(z)=H(z)X(z) \tag{3.41}$$

We can express

$$Y(z) = \frac{1}{M} \sum_{l=0}^{M-1} H(e^{-j 2\pi l / M} z^{1/M}) X(e^{-j 2\pi l / M} z^{1/M}) \tag{3.42}$$

Evaluating Y(z) in equation 3.42 on the unit circle, $z = e^{i\omega}$, leads to the result

$$Y(e^{j\omega'}) = \frac{1}{M} \sum_{l=0}^{M-1} H(e^{j(\omega'-2\pi l)/M}) X(e^{j(\omega'-2\pi l)/M}) \tag{3.43a}$$

where

$$w' = 2\pi fT' \text{ (in radians relative to sampling period T')} \quad (3.43b)$$

Equation (3.43a) expresses the Fourier transform of the output signal *y(m)* in terms of the transforms of the aliased components of the filtered input signal x(n). By writing the individual components of (3.43a) directly we see that

$$Y(e^{j\omega'}) = \frac{1}{M}\left[H(e^{j\omega'/M})X(e^{j\omega'/M}) + H(e^{j(\omega'-2\pi)/M})X(e^{j(\omega'-2\pi)/M})\right] + \ldots \quad (3.44)$$

The low-pass filter $H(e^{j\omega})$ is to filter x(n) so that its components above the frequency $\omega = n/M$ are negligible. In terms of (3.42) this implies that all terms for $l \neq 0$ are removed and if the filter $H(e^{j\omega})$ closely approximates the ideal response of (3.29) then (3.42) becomes

$$Y(e^{j\omega'}) \cong \frac{1}{M} X(e^{j\omega'/M}), \qquad for \qquad |\omega'| \leq \pi \quad (3.45)$$

3.6 Hearing Aid Implementation

A hearing aid application makes use of the over-sampling concept and consists of a sigma-delta analog-to-digital converter (ADC) followed by a decimation filter as given in figure 3.1. The hearing aid uses sigma-delta modulators for the analog section, which requires hardware operating at high sampling rates.

The sigma-delta modulators are designed to shape the noise away from the frequency band of interest. This oversampling technique is based on the use of a digital filter to prevent the quantization noise aliasing during decreasing sampling rates. Such a decimation task requires a high quality low-pass filter and a sampling rate converter that brings the sampling frequency down. The implementation of a decimation filter for hearing aid systems using DA is discussed. We have implemented comb-half band FIR-FIR structure for the hearing aid application.

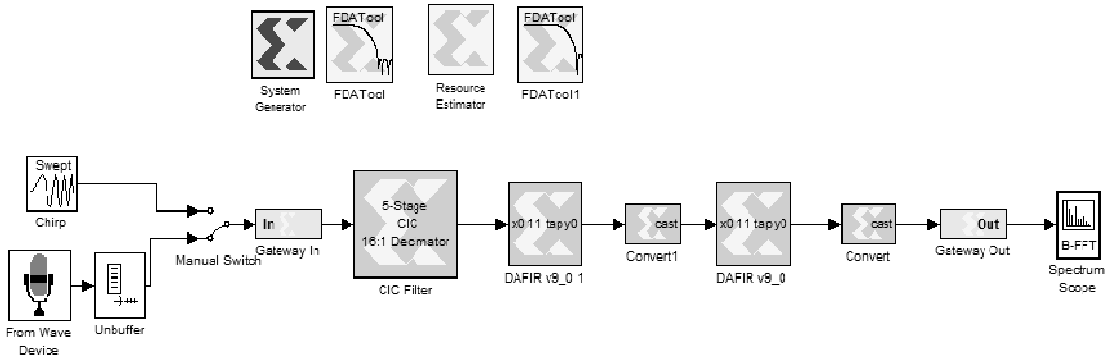


Figure 3. 15: Simulink block diagram of the decimation filter implemented

For the sake of comparison, we have designed and implemented a conventional comb-FIR-FIR decimation filter for the same specifications. The filter was implemented using Xilinx System-Generator and Matlab which is shown in figure 3.15. Input to the gatewayin (figure 3.14) is a sampled signal which was sampled at 1.28 MHz. The gatewayin converts the floating point number into equivalent fixed point number which is given as input to decimator. Decimator is followed by two FIR filters. The cast block is to convert the one fixed point number to another like a 16 bit fixed no to 8 bit fixed number and vice versa. This architecture is implemented using DA principle, which results in less hardware and less power consumption compared to other decimation filters.

3.6.1 Filter structure and design

A sigma-delta ADC generates a bit-stream at a rate of 1.28 MHz. This bit stream passes through a decimation filter and is down-sampled to a rate of 20 kHz. The specification of the decimation filter is mentioned in table 3.5. We have chosen a pass band of 4 kHz because the ear is sensitive to all sounds within 4 kHz [3, 69-71]. The filter has a pass band ripple of 0.001, which corresponds to the flat response of the filter. Since the transition band is a small percentage of the sampling rate, the filter will have many taps, and the zeros will be closer to each other causing the filter to be more sensitive to noise. The conventional decimation filter is implemented by this method, and the proposed system eliminates such problems by using a multistage multirate approach. We have implemented the decimation filter using a half-band FIR-FIR structure. The block diagram of the complete decimation filter is presented in figure 3.16.

Table 3. 5: Filter Specification

Decimation factor	16
Pass band frequency	4 kHz
Pass-band ripple	0.001
Cutoff frequency	15 kHz
Output word length	12 bits
Output word rate	20 kHz

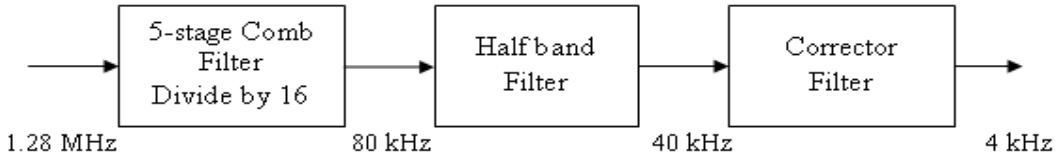


Figure 3. 16: Decimation filter

3.6.2 Implementation of the Comb Filter

The input of the decimation filter is generated from a sigma-delta modulator running at 1.28 MHz. At the sigma-delta modulator output, the out-band signals surrounding multiples of sampling frequency alias into the desired band after decimation occurs. Comb filter can be designed to present a notch at each of the frequencies that will alias to the base-band. Moreover, comb filter needs no multiplier and consists of simple operations suitable at high frequencies. The comb filter decimates by a factor M as given in equation (3.46).

$$H(z) = \frac{1}{M^k}\left(\frac{1-z^{-M}}{1-z^{-1}}\right)^k \tag{3.46}$$

The cascade of comb filters decimates the input signal by half the over sampling ratio to simplify the FIR filter computation. The comb filter decimates from 1.28 MHz to 80 kHz due to a decimation factor of M = 16 [62-64]. The cascaded combo filter with a decimation factor 16 is shown in figure 3.17.

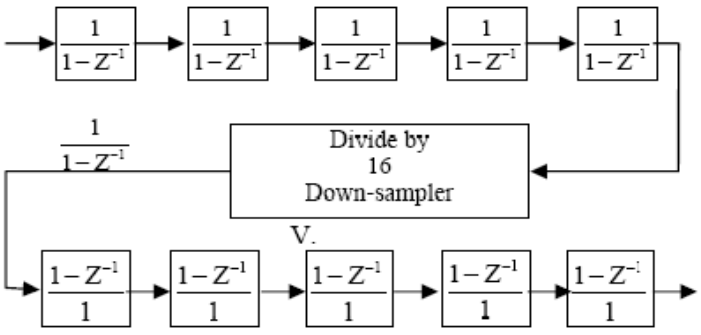


Figure 3. 17: Comb filter with decimation factor 16

A corrector filter can compensate distortion in pass band. The attenuation obtained in the current design of comb filter is about 65 dB as shown in figure 3.18.

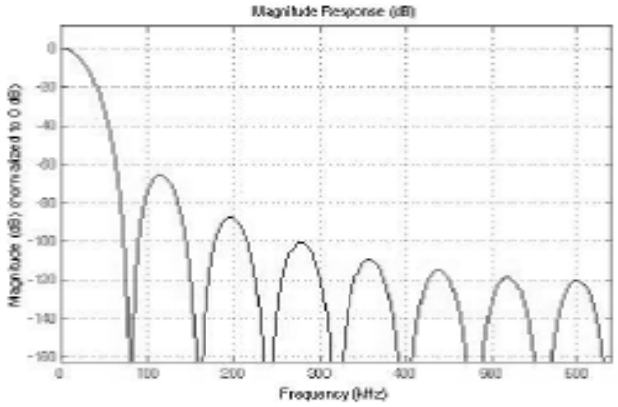


Figure 3. 18: Response of 5 stage comb filter

3.6.3 Half-Band Fir Filters Design

By considering the application requirements, FIR and IIR structures can be used to meet the design specifications. FIR filters offer great control over filter shaping and linear phase performance with waveform retention over the pass band [3, 67-70]. Due to its all-zero structure, the FIR filter has a linear phase response necessary for audio application, but at the expense of the high filter order. IIR filters, however, can be designed with much smaller orders than FIR filters at the expense of the nonlinear phase.

As it is very difficult to design a linear phase IIR filter, we have designed a half-band FIR filter for the same application using Remez algorithm as given in equation 3.47.

$$h(n) = \begin{cases} 0, & if \ \ n \neq \dfrac{N-1}{2} \\ 1, & if \ \ n = \dfrac{N-1}{2} \end{cases} \tag{3.47}$$

We have used a half-band filter; the number of taps is reduced considerably since the odd coefficients are zeros, which reduces the hardware and the power consumption. The half-band FIR filter has a transition band of about 15 kHz with the input sampling frequency of 80 kHz, a pass band of 20 kHz, a cut-off frequency of 35 kHz, and a stop band attenuation of -85 dB. The designed FIR filter has 11 coefficients and its response shown in figure 3.19.

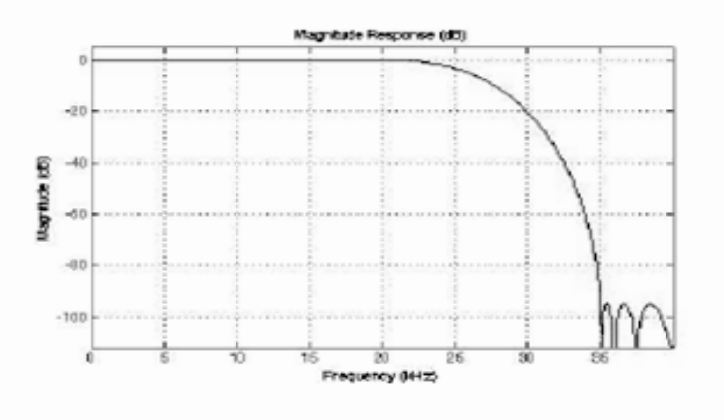


Figure 3. 19: Magnitude response of half band filter

3.6.4 Corrector Filter Design

An additional FIR filter is designed to push out of band undesired signals. The FIR filter is used in the last stage instead of a shaping filter for less power consumption because a shaping filter has more taps than an FIR filter [3, 69-70]. The designed FIR filter has a pass band of 4 kHz and a stop band of 15 kHz with a sampling frequency of 40 kHz. The human ear is very sensitive to all sounds within 4 kHz, due to which the designed filter has a pass band of 4 kHz. The output of the corrector filter has a sampling frequency of 20 kHz, which is ideal for the operation of the following signal processing stages. The response of the corrector filter is shown in figure 3.20. From the figure it can be seen that stop band attenuation of more than 100 dB is obtained which is suitable for this corrector filter.

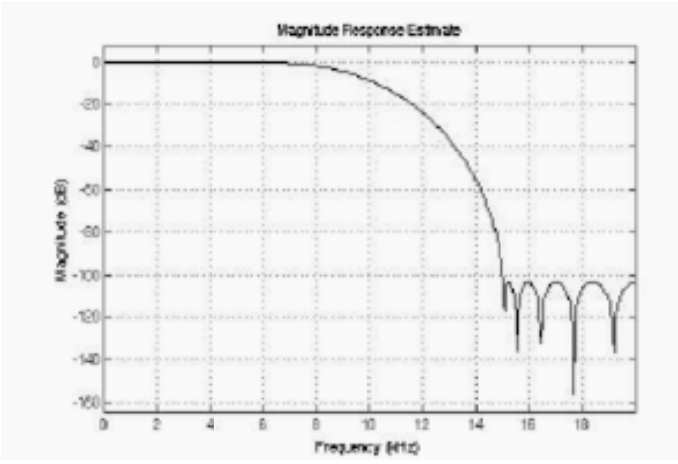


Figure 3. 20: Magnitude response of last stage corrector filter

3.7 Results & Conclusion

Table 3.6 shows the cell usage for the Comb-half-band FIR-FIR filter design and comb-FIR-FIR filter design. The comb-half band FIR-FIR filter design uses less

hardware compared to the comb-FIR-FIR filter design. The power consumption is less using comb-half band FIR-FIR filter design compared to the comb-FIR-FIR filter. Frequency used for calculating power consumption is 1.28 MHz. Table 3.6 shows that the proposed decimation filter using Comb-halfband-FIR-FIR architecture requires less hardware and contributes to hardware saving and a power saving compared to the comb-FIR-FIR architecture as shown in table 3.7. This filter is being implemented in Virtex-II pro board from Xilinx and the resource estimator from the system generator is to estimates the resources used by the design as given in table 3.6.

Table 3. 6: Area used

Cells used	slices	Slice flip flops	4-LUT	Logic	Shift registers	IO
Comb-FIR-FIR	695	1474	773	628	145	22
Comb-Half band FIR-FIR	639	1174	699	584	115	22

Table 3. 7: Power consumption

Decimation Filter Architecture	No. taps	Power Consumption
5-stage Comb-FIR-FIR	22	61 mW
5-stage Comb-Half-band FIR-FIR	11	50 mW

The comb-half-band FIR-FIR decimation filter is designed using Matlab and checked for real-time implementation using Simulink. The decimation filter is designed for 6-bit data stream input. The final output of the filter is 13 bits, and the stop band attenuation obtained is -65 dB. In addition, the distributed arithmetic multiplier is used for implementing in VHDL. The whole system was converted into VHDL code and then power was estimated in Xilinx ISE Power analyser tool as shown in table 3.7. Specifically, we compare the relative power consumption of two designs; the cell usage for each design is obtained using the synthesis report. The proposed decimation filter architecture requires less hardware and contributes to a hardware saving compared to the comb-FIR-FIR architecture.

Chapter-IV

An Adaptive Hearing Aid Algorithm using Booth-Wallace Tree Multiplier

4.1 Introduction

The decimation filter used in designing of a hearing aid has two major disadvantages which are

a. It requires more area for designing in FPGA.

b. As it is highly serial in nature, it requires more output latency.

Due to above reasons it consumes more power and we are specifically interested on lowering the power consumption of digital hearing aids. In this context we have used our own customized multiplier while maintaining the overall signal quality [76-77] to lower the power consumption.

Hearing impairment is often accompanied with reduced frequency selectivity which leads to a decreased speech intelligibility in noisy environments [72-75]. One possibility to alleviate this deficiency is the spectral sharpening for speech enhancement based on adaptive filtering [78-80] by which the intelligibility of the speech signal is maintained. Due to area constraints, such algorithms are usually implemented in totally time-multiplexed architectures, in which multiple operations are scheduled to run on a few processing units. This work discusses the power consumption in an FPGA implementation of the speech enhancement algorithm. It points out that power consumption can be reduced using Booth Wallace multiplier [82]. Several implementations of the algorithm, differing only in the degree of resource sharing are investigated aiming at power-efficiency maximization. At first an overview of the algorithm is given. Next the realized architectures using booth-Wallace tree multiplier are presented.

4.2 Wallace Tree Multiplier

Wallace trees are irregular in the sense that the informal description does not specify a systematic method for the compressor interconnections [81]. However, it is an efficient implementation of adding partial products in parallel [83]. The Wallace tree operates in three steps:

- Multiply: The multiplication is carried out using Booth Algorithm [82-85] which will generate n/2 partial product where n is number of bits of the multiplicand. The partial products are generated using the Booth recoding table given in table 4.1.

Table 4. 1: Booth recoding table

Mr_{i+1}	Mr	Mr_{i-1}	Recoded output
0	0	0	0
0	0	1	Y
0	1	0	Y
0	1	1	+2Y
1	0	0	-2Y
1	0	1	-Y
1	1	0	-Y
1	1	1	0

- Addition: As long as there are more than 3 wires with the same weights add a following layer. Take 3 wires of same weight and input them into a full adder. The result will be an output wire of same weight. If there are two wires of same weight, add them using half-adder and if only one is left, connect it to the next layer.
- Group the wires in two numbers and add in a conventional adder. A typical Wallace tree architecture is shown in figure 4.1 below. In the diagram AB0-AB7 represents the partial products.

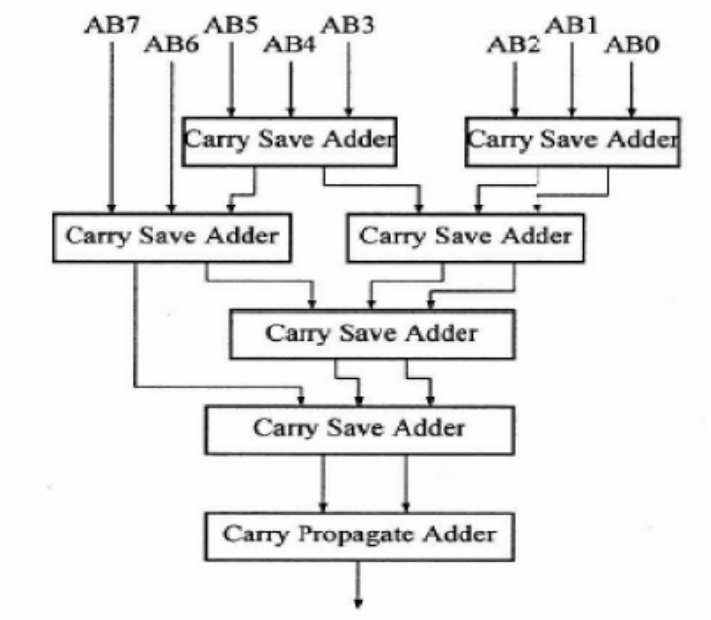


Figure 4. 1: Wallace tree multiplier

Wallace multipliers consist of AND-gates, Carry Save Adders and a Carry Propagate Adder or Carry Look-ahead Adder.

The n-bit CSA consists of disjoint full adders (FA's). It consumes three-bit input vectors and produces two outputs, i.e., n-bit sum vector S and n-bit carry vector *C*.

Unlike the normal adders [e.g. ripple-carry adder (RCA) and carry-look ahead adder (CLA)], a CSA contains no carry propagation. Consequently, the CSA has the same propagation delay as only one FA delay and the delay is constant for any value of n. For sufficiently large n, the CSA implementation becomes much faster and also relatively smaller in size than the implementation of normal adders. In Wallace multiplier carry save adders are used, and one carry propagate adder is used as shown in the figure 4.2. The basic idea in Wallace multiplier is that all the partial products are added at the same time instead of adding one at a time. This speeds up the multiplication process.

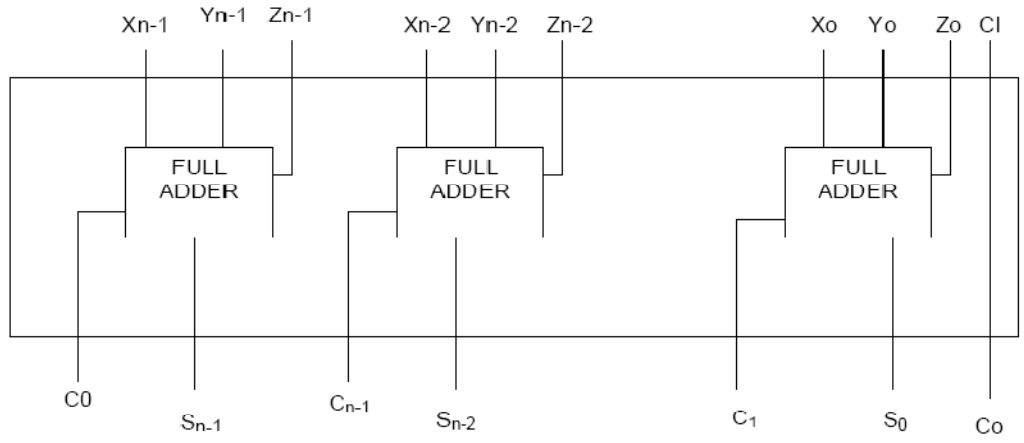


Figure 4. 2: Implementation of n bit CSA operation

4.2.1 Fast Adders

The final step in completing the multiplication procedure is to add the final terms in the final adder. This is normally called "Vector-merging" adder. The choice of the final adder depends on the structure of the accumulation array [84].

4.2.1.1 Carry Save Adder Tree (CSAT)

Carry Save Adder (CSA) can be used to reduce the number of addition cycles as well as to make each cycle faster. Figure 4.2 shows the implementation of the n-bit carry save adder. Carry save adder is also called a compressor. A full adder takes 3 inputs and produces 2 outputs i.e. sum and carry, hence it is called a 3:2 compressor. In CSA, the output carry is not passed to the neighbouring cell but is saved and passed to the cell one position down. In order to add the partial products in correct order, Carry save adder tree (CSAT) is used. In carry save adder (CSA) architecture, one adds the bits in each column of the first three partial products independently (by full adders). From there on, the resulting arrays of sum and carry bits and the next partial product

are added by another array of full adders [86]. This continues until all of the partial products are condensed into one array of sum bits and one array of carry bits. A fast adder (carry select or carry look-ahead) is finally used to produce the final answer. The advantage of this method is the possibility of regular custom layout. The disadvantage of the CSA method is the amount of delay of producing the final answer as the critical path is equivalent to first traversing all CSA arrays and then going through the final fast adder.

In contrast, in the Wallace tree architecture, all the bits of all of the partial products in each column are added together in parallel and independent of other columns. Then, a fast adder is used to produce the final result similar to the CSA method. The advantage of Wallace tree architecture is speed. This advantage becomes more pronounced for multipliers that are larger than 16 bits. However, building a regular layout becomes a challenge in this case. It can be seen that changing the Wallace tree multiplier into a multiplier/accumulator is quite simple. One needs to include the incoming data for accumulation in the set of partial products at the input of the Wallace tree section; and the Wallace tree will treat it as another Partial product. Also, merging multiple parallel multipliers and adders is as simple. It only needs to include all partial product bits in the same column in the inputs to the Wallace tree adders.

4.2.1.2 Carry Look Ahead Adder (CLA)

The concept behind the CLA is to get rid of the ripple carry present in a conventional adder design. The ripple of carry produces unnecessary delay in the circuit. For fast applications, a better design is required. The carry-look-ahead adder solves this problem by calculating the carry signals in advance, based on the input signals. It is based on the fact that a carry signal will be generated in two cases:

(a) when both bits Ai and Bi are 1, or

(b) when one of the two bits is 1 and the carry-in (previous carry) is 1.

For a conventional adder the expressions for sum and carry signal can be written as follows.

$$S = A \oplus B \oplus C \tag{4.1}$$

$$C = AB + BC + AC \tag{4.2}$$

It is useful from an implementation perspective to define S and Co as functions of some intermediate signals G (generate), D (delete) and P (propagate). G=1 means that a carry bit will be generated, P=1 means that an incoming carry will be propagated to C0. These signals are computed as

$$Gi = A_i.B_i \tag{4.3}$$

$$P_i = A_i \oplus B_i \tag{4.4}$$

We can write S and C_0 in terms of G and P.

$$C_0(G,P) = G + PC \tag{4.5}$$

$$S(G,P) = P \oplus C \tag{4.6}$$

Lets assume that the delay through an AND gate is one gate delay and through an XOR gate is two gate delays. It may be noted that the propagate and generate terms only depend on the input bits and thus will be valid after two and one gate delay, respectively. If one uses the above expression to calculate the carry signals, one does not need to wait for the carry to ripple through all the previous stages to find its proper value. Let's apply this to a 4-bit adder to make it clear.

$$C_1 = G_0 + P_0.C_0 \tag{4.7}$$

$$C_2 = G_1 + P_1.C_1 = G_1 + P_1.G_0 + P_1.P_0.C_0 \tag{4.8}$$

$$C_3 = G_2 + P_2.G_1 + P_2.P_1.G_0 + P_2.P_1.P_0.C_0 \tag{4.9}$$

$$C_4 = G_3 + P_3.G_2 + P_3.P_2.G_1 + P_3P_2.P_1.G_0 + P_3P_2.P_1.P_0.C_0 \tag{4.10}$$

Notice that the carry-out bit, C_{i+1}, of the last stage will be available after four delays (two gate delays to calculate the Propagate signal and two delays as a result of the AND and OR gate). The Sum signal can be calculated as follows,

$$S_i = A_i \oplus B_i \oplus C_i = P_i \oplus C_i \tag{4.11}$$

The Sum bit will thus be available after two additional gate delays (due to the XOR gate) or a total of six gate delays after the input signals Ai and Bi have been applied. The advantage is that these delays will be the same independent of the number of bits one needs to add, in contrast to the ripple counter. The carry look-ahead adder can be broken up in two modules: (i) the Partial Full Adder, which generates S_i, P_i and G_i and (ii) the Carry Look-ahead Logic, which generates the carry-out bits.

In the Booth-Wallace tree multiplier we have used CSA as it is a multi-bit operand adder and the delay in each stage of the multiplier is only delay of one full adder. But carry look-ahead adder is a two operand adder and can not be used as multi bit operand adder as carry propagates to next stage.

4.3 Adaptive lattice filter

The adaptive lattice filter [76-77, 88-90] consists of three parts and these are

a. Adaptive decorrelator
b. Analysis filter (lattice filter)
c. Synthesis filter(lattice structure)

4.3.1 The Adaptive Decorrelator

An adaptive filter is a filter that adjusts its transfer function according to an optimizing algorithm. Because of the complexity of the optimizing algorithms, most adaptive filters are digital filters that perform digital signal processing and adapt their performance based on the input signal used.

The adaptive process involves the use of a cost function, which is a criterion for optimum performance so that the filter coefficients adjusted to minimize the cost on the next iteration [3]. The block diagram for such filter is presented in figure 4.3 that serves as a foundation for particular adaptive filter realizations, such as Least Mean Squares (LMS), Recursive Least Squares (RLS) or steepest descent algorithm etc. The idea behind the block diagram is that a variable filter extracts an estimate of the desired signal.

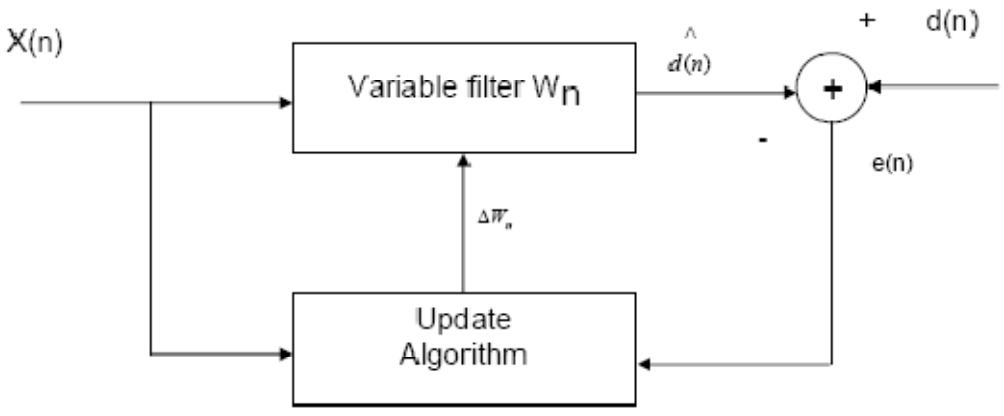


Figure 4. 3: Block diagram of an adaptive filter

The structure presented in figure 4.3 is described now. We take the following assumptions:

1. The input signal is the sum of a desired signal d(n) and interfering noise v(n)

$$x(n) = d(n) + v(n) \tag{4.12}$$

2. The variable filter has a Finite Impulse Response (FIR) structure. For such structures the impulse response is equal to the filter coefficients. The coefficients for a filter of order p are defined as

$$W_n(0)=[w_n(0),w_n(1),\ldots w_n(p)]^{T} \tag{4.13}$$

3. The error signal or cost function is the difference between the desired and the estimated signal.

$$e(n) = d(n) - \hat{d}(n) \tag{4.14}$$

The variable filter estimates the desired signal by convolving the input signal with the impulse response. In vector notation this is expressed as

$$d(n) = W^{T}(n)X(n) \tag{4.15}$$

where

$$x(n)=[x(n), x(n-1),\ldots\ldots..,x(n-p)]^{T} \tag{4.16}$$

is an input signal vector. Moreover, the variable filter updates the filter coefficients at every time instant

$$W_{n+1}=W_n+ \Delta W_n \tag{4.17}$$

where Δw_n is a correction factor for the filter coefficients. The adaptive algorithm generates this correction factor based on the input and error signals. LMS and RLS define two different coefficient update algorithms. The speech signal to be transmitted is spectrally masked by noise. By using an adaptive filter, we can attempt to minimize the error by finding the correlation between the noise at the signal microphone and the (correlated) noise at the reference microphone. In this particular case the error does not tend to zero as we note the signal d(k) = x(k) + n(k) whereas the input signal to the filter is x(k) and n(k) does not contain any speech [72]. Therefore it is not possible to "subtract" any speech when forming e(k)=d(k)-d(n) . Hence in minimising the power of the error signal e(k) we note that only the noise is removed and e(k) =~ x(k).

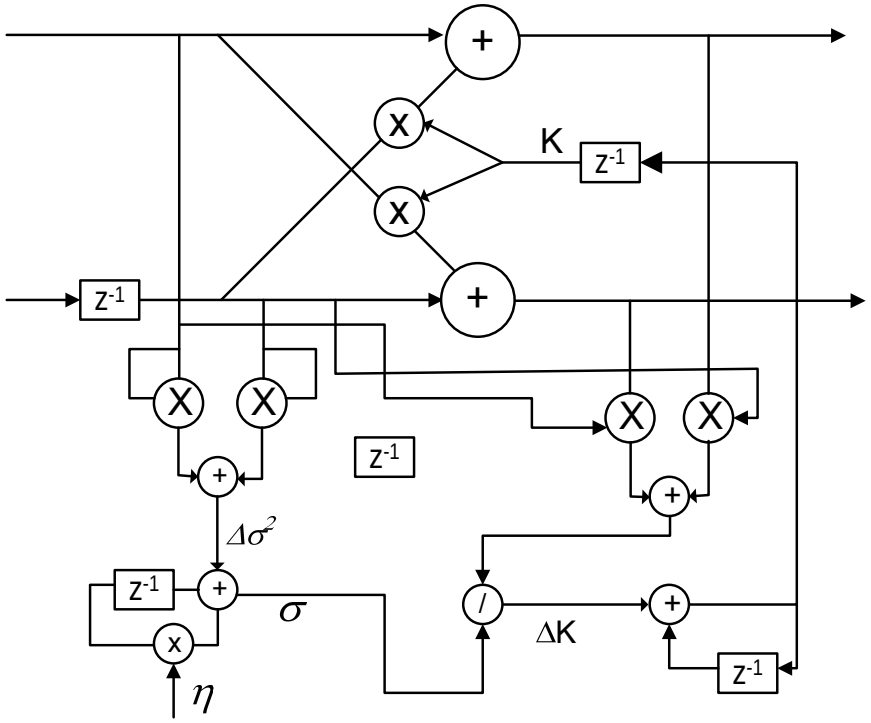


Figure 4. 4: Adaptive gradient lattice filter

Figure 4.4 depicts the structure of the adaptive gradient lattice decorrelator. The illustration shows three stages only with indices 1, i and m. good results typically require a filter order m where m can vary from 8 to 10 for speech sampled at 8 kHz. The output signal with vanishing autocorrelation is computed on the upper signal path by subtracting from the input sample suitable fractions of the signal values on the lower path. The multipliers $K_1, K_2 \ldots\ldots\ldots K_m$ are iteratively computed as in equation 4.18.

$$K_i[n] := K_i[n-1] + \Delta K_i[n] \qquad (4.18)$$

at every sampling interval n. the details of this process are illustrated in figure 4.4 for the i-th stage. Input and output values on upper and lower signal path to and from the i^{th} stage contribute to the computation of the update value ΔK_i.

Both output values are multiplied with the input values on the opposite path and then summed up to form the numerator in the subsequent computation of the up-date value. The denominator σ^2 is iteratively computed.

$$\sigma^2 := e.\ \sigma^2[n-1] + \Delta\ \sigma^2[n] \qquad (4.19)$$

The incremental value equals the sum of the two squared input values. The iterative computation of the denominator defines an exponentially decaying window which progressively decreases the influence of past contributions.

The computationally expensive division yields fast converging filter coefficients k_i independent of the varying input signal power level. This remarkable property is indispensable for good enhancement results. It is also clear contrast to simpler algorithms replacing the division by a multiplication with a small convergence constant $0<\mu<<1$. The longest delay through a string of lattice filters extends from the output of the storage element in the first lattice filter, through a multiplication with the first reflection coefficient, and then through an addition for each stage of the lattice filter until the output is produced in the final stage. For a large number of lattice filter stages, this longest delay can be reduced by a lattice filter optimization for speed which defers the final carry propagating addition until after the final lattice filter stage. This requires the transmission of an additional value between lattice filter stages. The multiplication process is speeded up using booth multiplier and the accumulation process is done faster using the Wallace multiplier.

4.3.2 The Analysis Filter

The analysis filter $H(z)=[1-A(z/\beta)]$ is illustrated in figure 4.5. Its structure [3, 78-80] is similar to that of the adaptive decorrelator shown in figure 4.4. The only difference is the multiplication with the filter parameter β following every shift element z^{-1} on the lower signal path. Furthermore, the analysis filter does not need a separate circuitry for coefficient update. It instead requires and therefore copies the filter coefficients $K_1, K_2 \ldots K_m$ computed by the unmodified ($\beta = 1$) filter structure, i.e., the adaptive decorrelator.

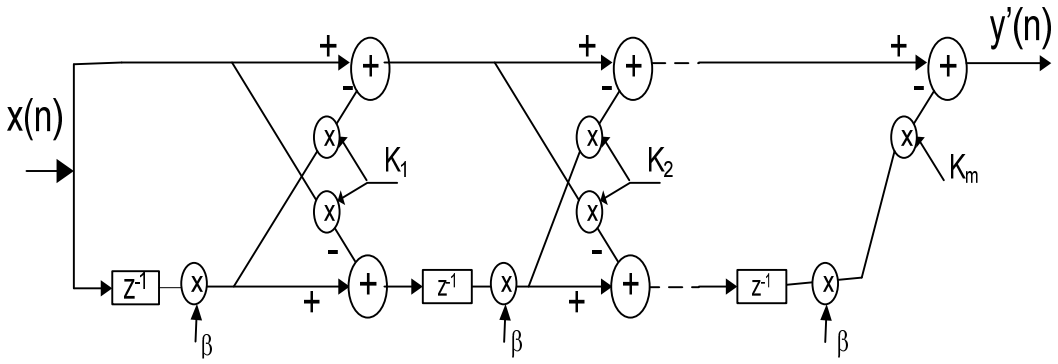


Figure 4. 5: Analysis filter (1-A(z/β), x(n) is input and y'(n) is output

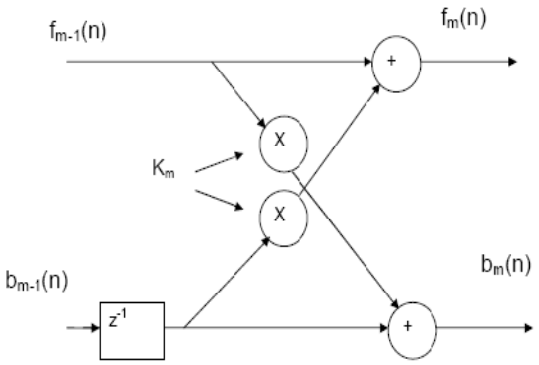


Figure 4. 6: Single stage of analysis filter

The two mathematical equations for the single stage analysis filter as shown in figure 4.6 are

$$f_m(n) = f_{m-1}(n)-k_m b_{m-1}(n-1) \quad (4.20)$$

$$b_m(n) = b_{m-1}(n-1)-k_m f_{m-1}(n) \quad (4.21)$$

Some characteristics of the Lattice predictor are

- It is the most efficient structure for generating simultaneously the forward and backward prediction errors.
- The lattice structure is modular: increasing the order of the filter requires adding only one extra module, leaving all other modules the same.
- The various stages of a lattice are decoupled from each other in the following sense: The memory of the lattice (storing $b0(n ¡ 1); : : : ; bm¡1(n ¡ 1)$) contains orthogonal variables, thus the information contained in $x(n)$ is splitted in m pieces, which reduces gradually the redundancy of the signal.
- The similar structure of the lattice filter stages makes the filter suitable for VLSI implementation.

Lattice filters typically find use in such applications as predictive filtering, adaptive filtering, and speech processing. One desirable feature of lattice filters are their use of reflection coefficients as the filter parameter. Algorithms exist to compute reflection coefficients to obtain the optimal linear filter for a given filter order. Reflection coefficients have the additional property that for some applications, the optimal reflection coefficients remain unchanged when going from a lower order filter to a

higher order filter. Thus, when adding additional filter stages, only the reflection coefficients for the added stages need to be computed. The straight forward implementation of a lattice filter would be composed of two multipliers, two adders and a latch per lattice filter stage as shown in figure 4.6. The longest timing path to the output of a string of lattice filters starts at the multiply of the first lattice filter stage and propagates through the adders of each lattice filter stage. The delay for an n stage conventional lattice filter is equal to the delay of one multiply and n CLAs. The modification with filter parameter β causes the analysis filter to produce an output signal with reduced formants instead of a signal with completely flat spectral envelope as produced by the adaptive decorrelator.

4.3.3 The Synthesis Filter

When considering IIR filters, the direct form filter is the common structure of choice. This is true, in general, because when designing an algorithm which adapts the parameters a_k and b_k, the coefficients of the difference equation, described below, are manipulated directly.

$$Y_k+a_1Y_{k-1}+- - - - - - - - - - - - +a_mY_{k-m}=b_0U_k+b_1U_{k-1}+- - - - - - +b_mU_{n-m} \qquad (4.22)$$

Some problems exist in using the direct form filter for adaptive applications. First of all, ensuring stability of a time-varying direct form filter can be a major difficulty. It is often computationally a burden because the polynomial, A(z), made up of the a_k parameters, must be checked to see if it is minimum phase at each iteration. Even if the stability was assured during adaptation, round off error causing limit cycles can plague the filter. Parallel and cascade forms are often used as alternatives for direct form filters. These consist of an interconnection of first and second order filter sections, whose sensitivity to round off errors tends to be less drastic than for the direct form filter. Since the filter is broken down into a factored form, the round off error associated with each factorization only affects that term. In the direct form filter, the factors are lumped together so that round off error in each term affects all of the factors in turn.

A larger problem exists for both parallel and cascade forms: the mapping from transfer function space to parameter space is not unique. Whenever the mapping from the transfer function space to the parameter space is not unique, additional saddle points in the error surface appear that would not be present if the mapping had been

unique. The addition of these saddle points can slow down the convergence speed if the parameter trajectories wander close to these saddle points. For this reason, these filter forms are considered unsuitable for adaptive filtering.

A tapped-state lattice form has many of the desirable properties associated with common digital filters and avoids the problems discussed above. Due to the computational structure, the round off error in this filter is inherently low.

Direct implementation of the IIR filter can lead to instabilities if it is quantized. The filter is stable using the following structure [3, 78-81, 86-87,91]. The structure of the synthesis filter $H(z)=[1-A(z/\gamma)]^{-1}$ is shown in figure 4.7. The synthesis filter also requires and copies the filter coefficients $K_1, K_2,\ldots\ldots K_m$ from the adaptive decorrelator at every sampling interval. The structure in figure 4.7 also shows a synthesis filter modified by the multiplication with the filter parameter γ succeeding every shift element z^{-1} on the lower signal path. The unmodified synthesis filter ($\gamma=1$) restores the original formants in the output when a signal with flat spectral envelope is fed to its input. The modification with a parameter value less than unity causes the synthesis filter to produce an output signal with partially restored formants only. The spectral sharpening effect results from a suitable choice of both filter parameters $0<\beta<\gamma<1$. Experiments with one adaptive filter only failed in producing satisfactory speech enhancement results.

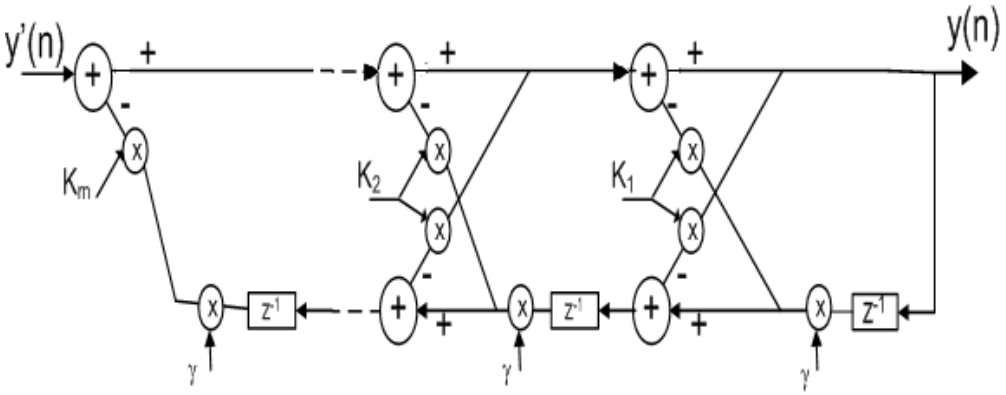


Figure 4. 7: Synthesis filter, y'(n) input and y(n) is output

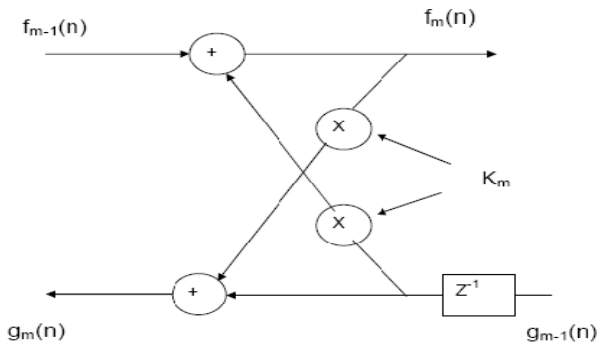


Figure 4. 8: Single stage of sysnthesi filter.

The two mathematical equations for the single stage synthesis filter are shown below.

$$f_m(n) = f_{m-1}(n)-k_m b_{m-1}(n-1) \quad (4.23)$$

$$g_m(n) = g_{m-1}(n-1)-k_m f_{m-1}(n) \quad (4.24)$$

The computational complexity of a digital filter structure is given by the total number of multipliers and the total number of two input adders required for its implementation which roughly provides an indication of its cost of implementation. The synthesis filter is stable if the magnitudes of all multiplier coefficients in the realization are less than unity i.e. $-1<K_m<1$ for m=M, M-1,M-2, …1,0.

4.4 Hearing Aid Design

4.4.1 Spectral Sharpening For Speech Enhancement

Speech enhancement usually results from adaptively filtering the noise reference signals and subsequently subtracting them from the primary input. However, a procedure for speech enhancement based on a single audio path is presented here. It is therefore applicable for real world situations. An example of such a situation is using hearing aid equipment. The hearing impaired person could place additional microphones close to noise sources only rarely. Current hearing aid equipment are used for filtering and amplifying the speech signal, this suggests that hearing impairment is just a more or less reduced sensitivity to sound pressure in various frequency intervals. This view however neglects the loss of frequency discrimination which can be efficiently compensated by the spectral sharpening technique presented. The idea of spectral sharpening originates from the adaptive post filtering method in modern speech coding schemes at bit rates around 8 kb/s and lower [85]. With these algorithms speech is encoded segment by segment. The linear prediction filter is any way computed in every speech segment for the encoding process as

$$A(z)=a_1z^{-1} + a_2z^{-2}+.....+ a_mz^{-m} \quad (4.25)$$

and post- filtering with the transfer function

$$H(z)=\frac{1-A(z/\beta)}{1-A(z/\gamma)} \quad (4.26)$$

and constant filter parameters $0<\beta<\gamma<1$ is subsequently performed with a moderate computational increase.

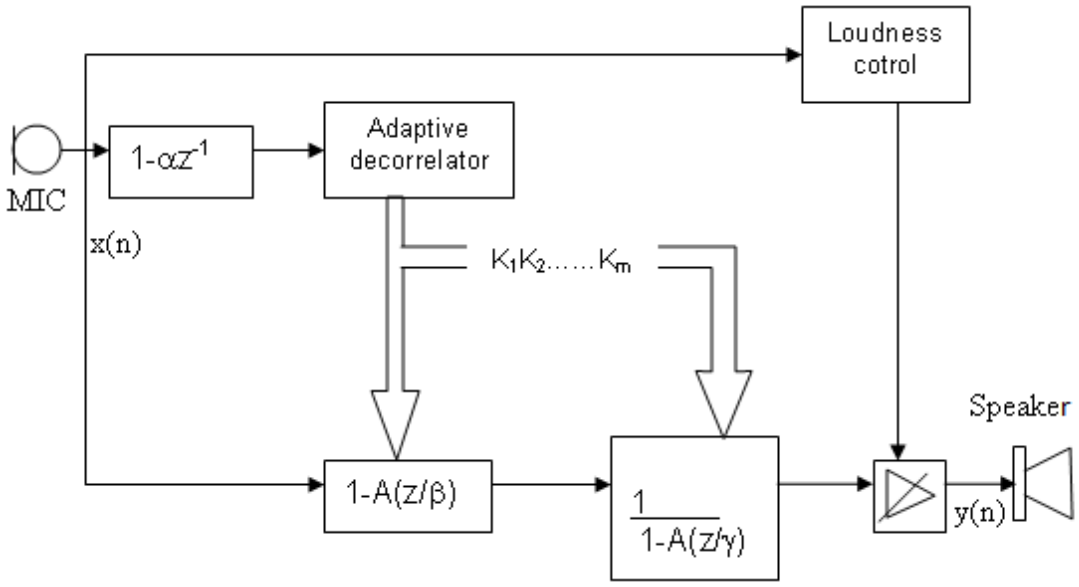


Figure 4. 9: Block diagram of spectral sharpening for speech enhancement.

Figure 4.9 shows the block diagram of spectral sharpening of speech sharpening [78-80] for speech enhancement. The speech signal x[n] from the microphone splits into three distinct paths. The signal on the lowest path passes through the analysis filter $[1-A(z/\beta)]$ and subsequently through the synthesis filter $[1-A(z/\gamma)]^{-1}$. Both filters are implemented as lattice filters with the analysis and synthesis structures respectively. They both require the identical set of reflection coefficients $K_1, K_2,.....K_m$. where m represents the number of stages which is updated in every sampling interval by the adaptive decorrelator shown on the middle path of figure 4.3. The filter parameters β and γ do not vary with time.

A high pass filter $1-\alpha z^{-1}$ is shown in front of the adaptive decorrelator, where x=1 may be chosen for simplicity. The high pass filter is used in order to compensate the spectral tilt of natural speech: the average power of the speech signal decreases above 1 KHz at a rate of ~ 10 db per octave. The adaptive transfer function in equation (4.26) enhances this spectral tilt even more when the filter coefficients

K_1,K_2,……..K_m are computed from the speech signal x[n] directly. Efficient speech enhancement requires however that the various formants are more or less uniformly emphasized, regardless of their relative power level. This is possible with the use of the high pass filter. It compensates at least partially the original spectral tilt.

The decorrelator on the middle signal path of the figure is an adaptive gradient lattice filter. It produces an output signal with vanishing autocorrelation by updating its filter coefficients in every sampling interval to the continuously changing input signal characteristics. The output signal is not required in this application, however. The updated filter coefficients K_1, K_2……K_m are of interest only for the use in the analysis and synthesis filter.

4.4.2 Spectral Sharpening for Noise Reduction

The block diagram of the spectral sharpening process for noise reduction is illustrated in figure 4.10. The arrangement of adaptive decorrelator, analysis and synthesis filters agrees with the previous block diagram in figure 4.9, however there various differences like

1. no loudness control,.
2. the input signal x[n] goes directly to the adaptive decorrelator, and
3. a high pass filter precedes the analysis and synthesis filters.

$$H_{hp}(z) = \frac{b(1 - z^{-1})}{1 - az^{-1}} \tag{4.27}$$

The reasons for these differences are as follows.

As mentioned in the previous section the spectral sharpening process

$$H(z) = \frac{1 - a(z/\beta)}{1 - a(z/\gamma)} \tag{4.28}$$

introduces a signal dependent amplification, signal segments with strong formant structure are amplified more than segments with a rather flat spectral envelop. In the sequel it is assumed that back ground noise is the major source for signal degradation and that its spectrum reveals relatively flat resonances only. Speech segments with strong resonances clearly profit in this situation. They experience a remarkable amplification compared to noisy segments. The loudness compensation of the previous block diagram is consequently omitted in order to preserve this effect.

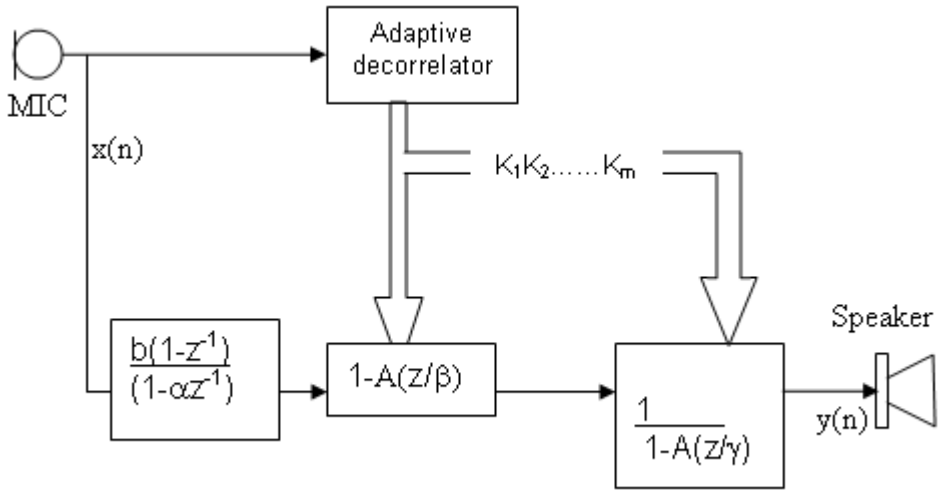


Figure 4.10: Block diagram of spectral sharpening by noise reduction

Best results require that the input signal is directly fed to the adaptive decorrelator. Only negligible amplification is then applied to noisy signal segments as a consequence of their assumed approximately flat spectrum [89, 91]. The spectral sharpening process further enhances the spectral tilt of speech when the filter parameters are estimated from the speech signal without prior compensation.

The high pass filter which preceded the adaptive decorrelator in the figure 4.9 has been shifted to the bottom signal path in figure 4.10 in order to avoid the scheme from producing a dull sound.

4.4.3 High Pass Filter

In signal processing, there are many instances in which an input signal to a system contains extra unnecessary content or additional noise which can degrade the quality of the desired signal. In such cases we may remove or filter out the useless samples. For example, in the case of the telephone system, there is no reason to transmit very high frequencies since most speech falls within the band of 700 to 3,400 Hz. Therefore, in this case, all frequencies above and below that band are filtered out. The frequency band between 700 and 3,400 Hz, which isn't filtered out, is known as the pass band, and the frequency band that is blocked out is known as the stop band. FIR, Finite Impulse Response, filters are one of the primary types of filters used in Digital Signal Processing [3]. FIR filters are said to be finite because they do not have any feedback. Therefore, if an impulse is sent through the system (a single spike) then the output would invariably become zero as soon as the impulse runs through the filter.

There are a few terms that are used to describe the behaviour and performance of FIR filter. These are

- Filter Coefficients - The set of constants, also called tap weights, used to multiply against delayed sample values. For an FIR filter, the filter coefficients are, by definition, the impulse response of the filter.
- Impulse Response – A filter's time domain output sequence when the input is an impulse. An impulse is a single unity-valued sample followed and preceded by zero valued samples. For an FIR filter the impulse response of a FIR filter is the set of filter coefficients.
- Tap – The number of FIR taps, typically N, tells us a couple things about the filter. Most importantly it tells us the amount of memory needed, the number of calculations required, and the amount of "filtering" that it can do. Basically, the more taps in a filter results in better stop band attenuation (less of the part we want filtered out), less ripple (less variations in the pass band), and steeper roll off (a shorter transition between the pass band and the stop band).
- Multiply-Accumulate (MAC) – In the context of FIR Filters, a "MAC" is the operation of multiplying a coefficient by the corresponding delayed data sample and accumulating the result. There is usually one MAC per tap.

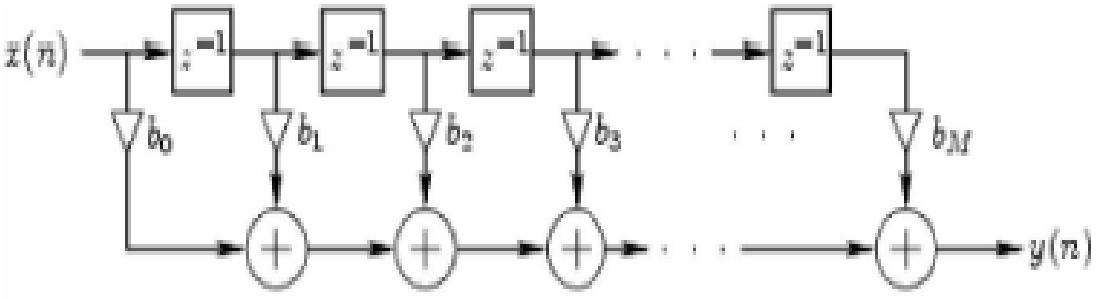


Figure 4. 11: General causal FIR filter structure

Figure 4.11 gives the signal flow graph for a general finite-impulse-response filter (FIR). Such a filter is also called a transversal filter, or a tapped delay line. The implementation is one example of a direct-form implementation of a digital filter. The impulse response h(n) is obtained at the output when the input signal is the impulse

signal δ=[1 0 0 0…]. If the kth tap is denoted b_k, then it is obvious from figure 4.11 above that the impulse response signal is given by

$$h(n)=\begin{cases}0, & n<0\\ b_n, & 0\le n\le M\\ 0, & n>M\end{cases} \tag{4.29}$$

In other words, the impulse response simply consists of the tap coefficients, pretended and appended by zeros.

Convolution Representation of FIR filters:

It may be noted that the output of the kth delay element in figure 4.11 is x(n-k),k=0,1,2…..m , where x(n) is the input signal amplitude at time n. The output signal y (n) is therefore

$$Y(n)=b_0x(n)+b_1x(n-1)+b_2x(n-2)+.....+b_M\,x(n-m)$$

$$=\sum_{m=0}^{M} b_m x(n-m)\;=\sum_{m=-\infty}^{\infty} h(m)x(n-m)=h(n)*x(n) \tag{4.30}$$

Where we have used the convolution operator '*' to denote the convolution of h(n) and x(n), as defined in above equation. An FIR filter thus operates by convolving the input signal x(n) with the filter's impulse response h(n).The transfer function of an FIR filter is given by the z transform of its impulse response.

$$H(z)=\sum h(n)z^{-n}=\sum_{n=0}^{M} b_n z^{-n} \tag{4.31}$$

Thus, the transfer function of every length N=M+1 FIR filter is an M^{th} order polynomial in Z.

The order of a filter is defined as the order of its transfer function. It may be noted from figure 4.11 that the order is also the total number of delay elements in the filter. When the number of delay elements in the implementation is equal to the filter order, the filter implementation is said to be canonical with respect to delay. It is not possible to implement a given transfer function in fewer delays than the transfer function order, but it is possible (and sometimes even desirable) to have extra delays.

Figure 4.12 shows the magnitude response of a FIR high pass filter with cutoff frequency of 700 Hz. The sampling frequency considered here is 8000 Hz. For exam-

ple the following figure 4.12-4.14 presents the simulation of high pass filter with 5 coefficients.

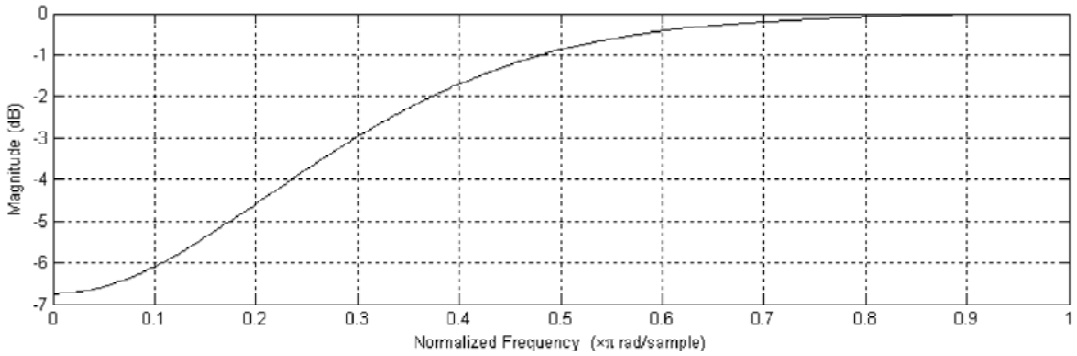


Figure 4. 12: Magnitude response of an high pass FIR filter (cut off frequency 700HZ)

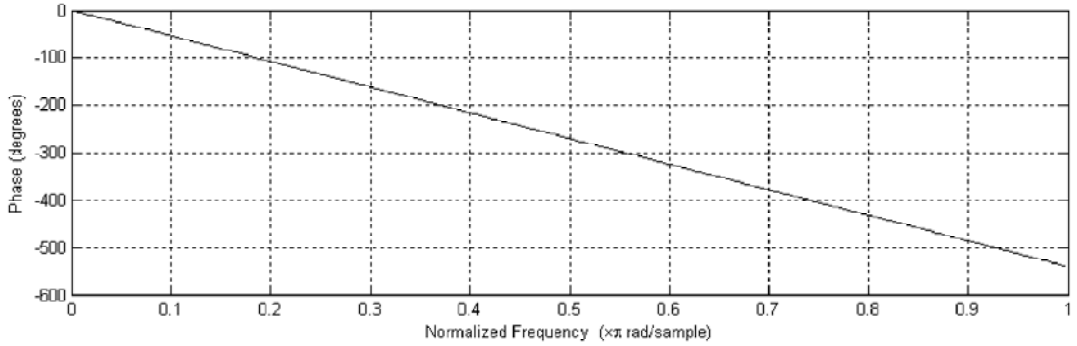


Figure 4. 13: Phase response of a high pass FIR filter (cut off frequency 700HZ).

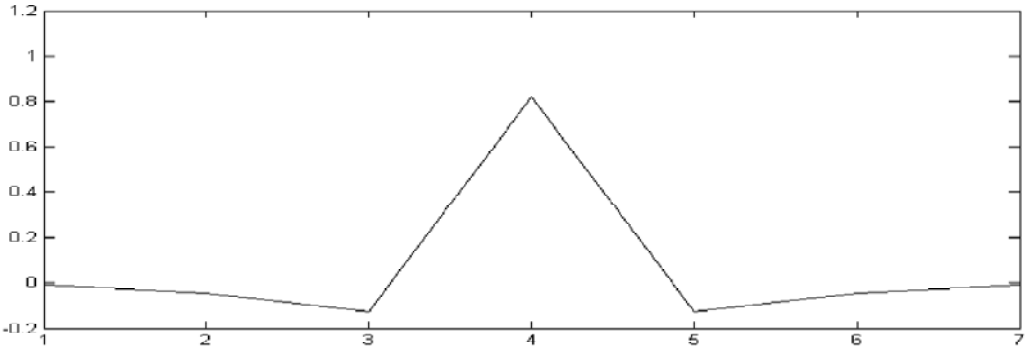


Figure 4. 14: Impulse response of a high pass FIR filter (cut off frequency 700HZ).

4.5 Experimental results and conclusion

4.5.1 Spectral Sharpening for Speech Enhancement – Simulation Result

Validation of the proposed filter was conducted using simulation tool. MATLAB v7.01 was used as platform. Speech signal constituting of 26000 samples at 8k samples/sec was generated using wave recorder of windows and captured as a Matlab data file. This constitutes 3.25s of voice data. The waveform generated is presented in

figure 4.15. The peak value of the signal generated is 275 mV. This signal formed the input to the hearing aid appliance. The output of a single stage is presented in figure 4.16 for β=0.04, γ=0.6 and μ=0.98. From the observation of the filter output it is seen that the output amplitude is nearly 600 mV. The single stage output with the filter parameters, β=0.4, γ=0.6 and μ=0.98 is presented in figure 4.17. In this case, peak amplitude is 390 mV which constitute a gain less than 2.

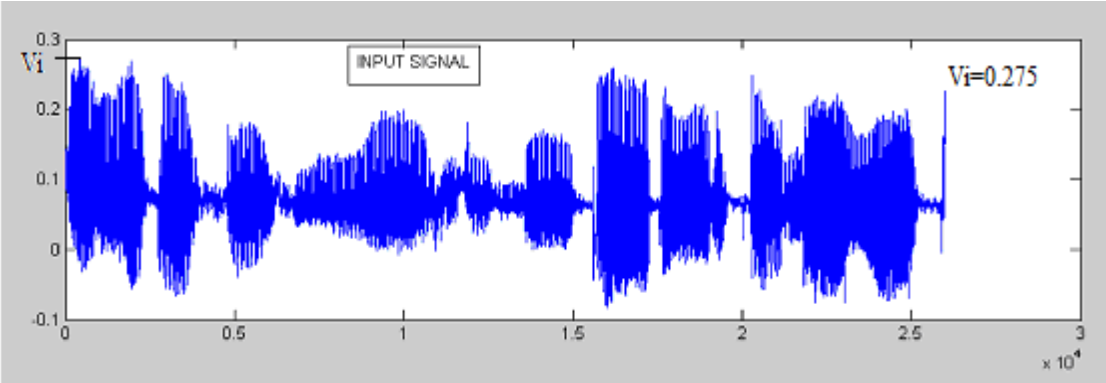


Figure 4. 15: Waveform of the 26000 sample speech input

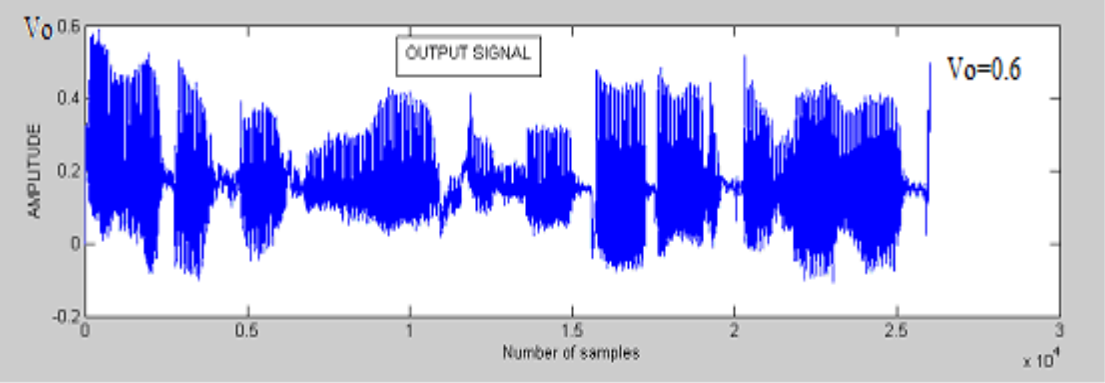


Figure 4. 16: Waveform of the 26000 sample hearing aid output using parameters β=0.04,γ=0.6,μ=0.98

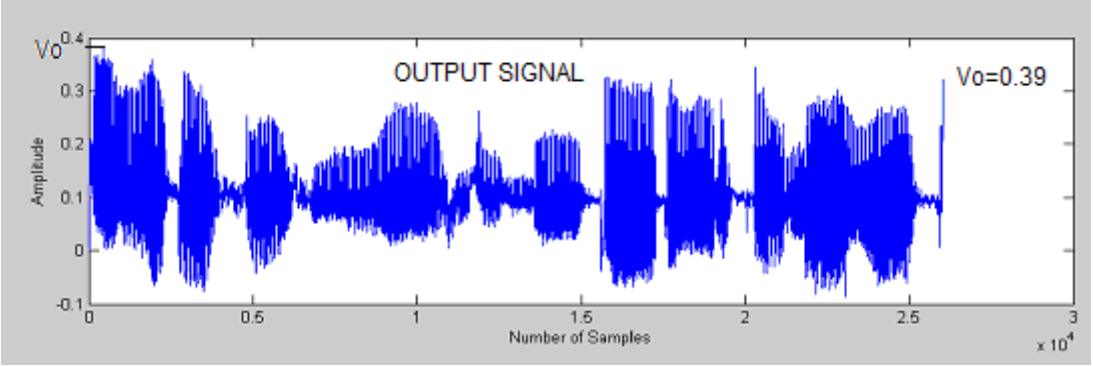


Figure 4. 17: Waveform of the 26000 sample hearing aid output using parameters β=0.4, γ=0.6, μ=0.98

Following this; the performance of a 8 stage filter is observed. The filter output for β=0.04, γ=0.6, μ=0.98 and β=0.4, γ=0.6 and μ=0.98 are presented in figure 4.18 and figure 4.19 respectively. From figure 4.15 and figure 4.18, it is seen that output is more than double. Considering the superior performance of the 8 stage filter output over single stage filter, a8 stage is used for hardware implementation.

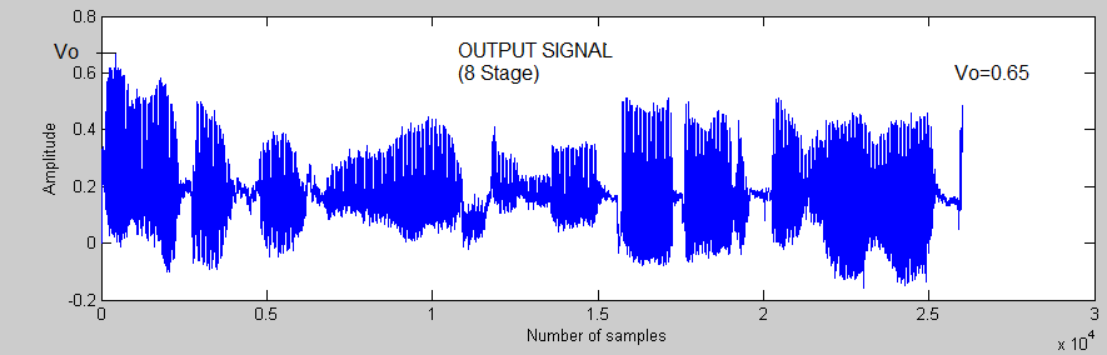


Figure 4. 18: Waveform of the 26000 sample hearing aid output using parameters β=0.04,γ=0.6,μ=0.98.

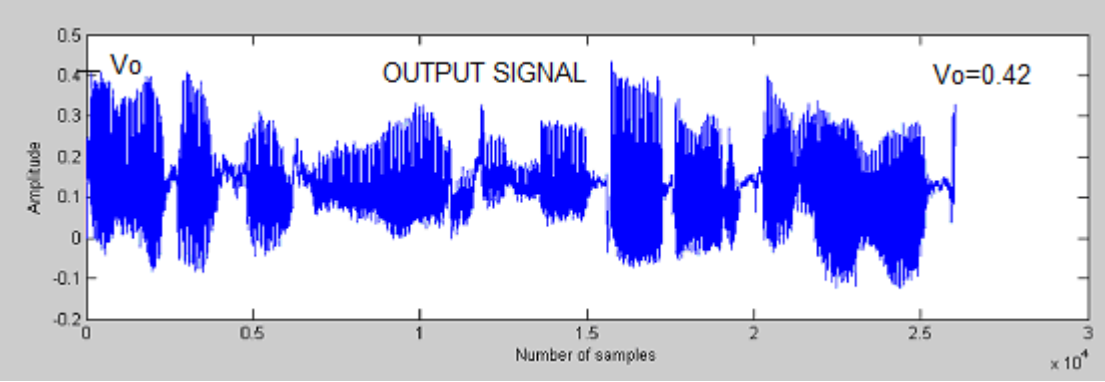


Figure 4. 19: Waveform of the 26000 sample hearing aid output using parameters β=0.4,γ=0.6,μ=0.98

4.5.2 FPGA based SIMULATION RESULTS.

4.5.2.1 Multipliers

The table below compares the cell usage of the three multipliers (SHIFT/ADD, Booth's and Booth-Wallace multiplier) for 8 bit by 8 bit multiplication and 16 bit by 16 bit multiplication. From the table we can see that the booth Wallace multiplier uses less hardware compared to that of the shift/add multiplier and booth multiplier. The details are given table 4.2.

Table 4. 2: Cell used for the three Multipliers in virtex2p

Cell Usage	Shift/add multiplier (8x8)	Shift/add multiplier (16x16)	Booth multiplier (8x8)	Booth multiplier (16x16)	Booth Wallace multiplier (8x8)	Booth Wallace multiplier (16x16)
BELS	240	1000	333	975	167	697
LUT-1	1	1	0	0	0	0
LUT-2	14	1	37	36	5	9
LUT-2	34	186	28	66	51	234
LUT-4	74	290	116	399	83	328
MUXCY	56	240	64	228	0	0
MUXF5	11	27	14	2	28	126
XORCY	49	225	61	219	0	0

Table 4. 3: Power consumption and Delay for two multipliers with 8x8 bits

Cells used	Slices	4-LUT	IO	Delay	Power consumption
Booth multiplier	109	192	32	24.41 ns	5 mW
Booth Wallace multiplier	76	139	32	20.62 ns	3 mW

From the table 4.3 we can see that the using the booth Wallace multiplier consumes less power compared to the booth multiplier and also that booth Wallace multiplier is faster than booth multiplier. Hence the Booth Wallace multiplier [82, 86, 88] is used for hearing aid design in VHDL in this investigation. The table 4.4 gives area used by the two multipliers. From this it can be seen that the booth Wallace tree multiplier uses less hardware than other.

Table 4. 4: Cell usage for hearing aid component in virtex2p

Cells used	Slices	Slice flip flops	4-LUT	Logic	Shift registers	IO
Booth multiplier	2684	183	5003	4979	24	32
Booth Wallace Multiplier	2583	196	4885	4866	19	32

4.5.2.2 Spectral Sharpening for Speech Enhancement in vhdl

The amplitude values of the speech signal sampled at 8 kS/s is rounded to 8 bits and stored in a text file for VHDL simulation. The hearing aid is designed in VHDL and is tested using different multipliers. The first 250 samples are taken as input for the hearing aid in VHDL .The output obtained through simulation is stored in a text file. The text file is read in MATLAB and is plotted as shown in the figure 4.22. The parameters used in VHDL are β=0.04,γ=0.6, μ=0.98.

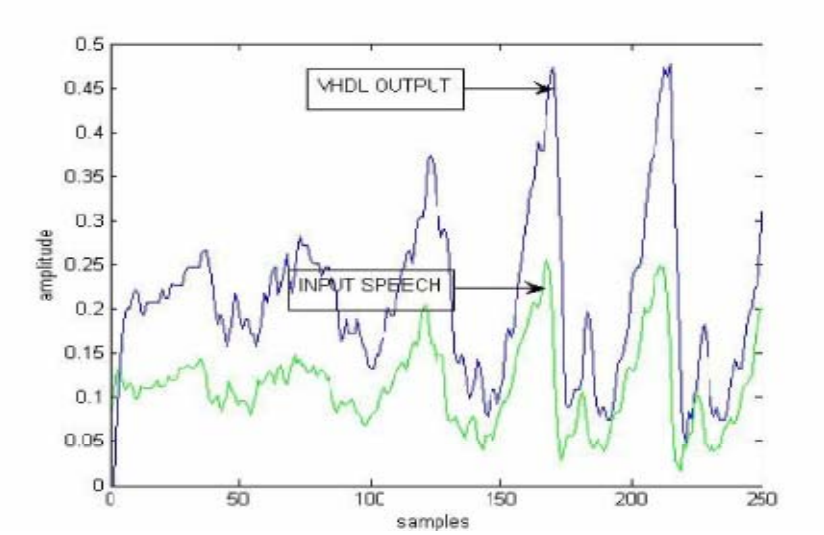


Figure 4. 20: Comparision of input speech signal with output using vhdl for 250 samples

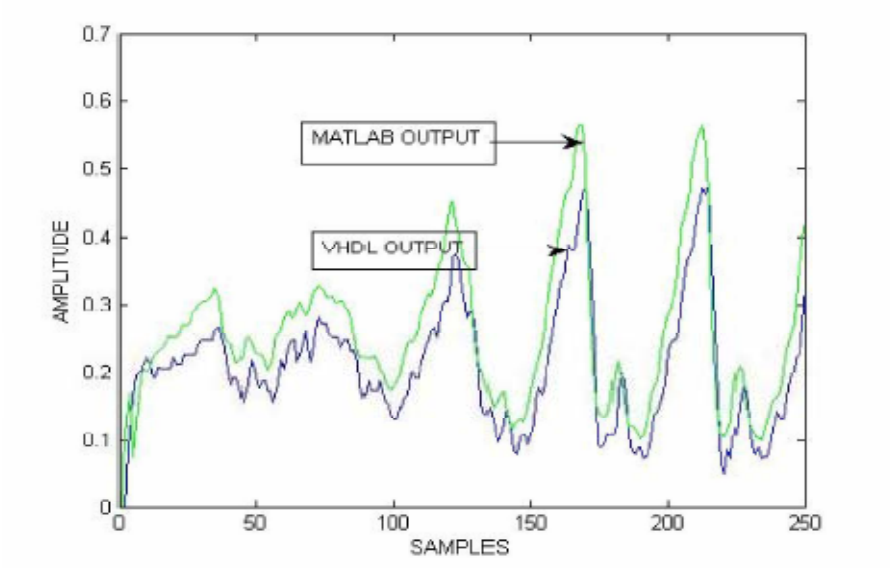


Figure 4. 21: Comparison of the MATLAB output speech signal with the output obtained using VHDL for 250 samples

From the figure we can see that the output obtained using VHDL is slightly less in magnitude than MATLAB output. This is due to rounding of the values and

due to fixed point multiplication. But from the figure 4.23 we see that the VHDL output follows the MATLAB output. The table below shows the resource utilization summary and power consumed by the two designs using different multipliers.

Table 4. 5: FPGA resources used and power of hearing aid design

Design name	Slice used out of 3008	4-LUTs used out of 6016	Slice FFs used out of 6016	Shift registers	Logics	IO used out of 348	Power consumption
High pass filter using Booth multiplier	241 (7%)	446 (7.4%)	41 (<1%)	-	-	25 (7%)	-
Synthesis filter	175 (5.8%)	323 (5.3%)	32 (<1%)	-	-	41 (11%)	-
Decorrelator	164 (5.4%)	292 (4.9%)	104 (1.7%)	-	-	40 (11%)	-
Hearing aid using Booth multiplier	2684 (89%)	5003 (83%)	183 (3%)	24	4979	32 (9.5%)	40 mW
Hearing aid using Booth Wallace multiplier	2583 (85%)	4885 (81%)	196 (3.3%)	19	4866	33 (9.5%)	30 mW

4.5.3 Conclusions

The speech enhancement system as depicted in figure 4.9 is implemented using Booth-Wallace tree multiplier and power calculation of the entire system is done using Xilinx Xpower Analyser. From the figure 4.21 it can be seen that FPGA based design results match with the Matlab output. Also referring to table 4.5 we can see that the power consumed by the hearing aid with Booth Wallace tree multiplier is less than the hearing aid using Booth multiplier which is about 25% less. So it can be inferred that the hearing aid using Booth Wallace tree multiplier is a better design on low power perspective. Furthermore, in terms of slice count the hearing aid using Booth-Wallace multiplier scores over hearing aid using Booth multiplier (uses 4 % less slices) that is evident from FPGA resource used by two different architectures. Therefore, Booth-Wallace tree multiplier is a better candidate compared to Booth multiplier while choosing multiplier architecture for hearing aid design.

Chapter-V

Low Power Filter Design using a Novel Dual Edge Triggered Latch

5.1 Introduction

In the chapters III and IV we have designed hearing aids using Decimation filters design using Distributed Arithmetic and Adaptive hearing aid using Booth-Wallace multiplier respectively. We find the performance of the adaptive filter is better than the decimation filter. In both the cases our focus is on reduction of power.

It has been seen that only 40% power is reduced in case of the adaptive filter when compared to decimation filter. We have so far concentrated mainly on algorithmic approach of power reduction. Now we proceed to integrate both circuit level and algorithmic level power reduction technique to apply to our problem of hearing aid design. In this chapter we focus on FIR filter which is the primary component of hearing aid design. Here in this design FDF (Folded Direct Form) of FIR filter structure is adopted. We have additionally focused on the clocking strategy to reduce glitches that facilitates power reduction; a novel circuit for latch is also proposed. This clocking strategy in conjunction with the latch is adapted for the FIR filter structure that is used for hearing aid.

5.2 FIR Filters

Finite impulse response (FIR) filtering is one of the most commonly used DSP functions. It is achieved by convolving the input data samples with the desired unit impulse response of the filter [3]. The output y(n) of an N tap FIR filter is given by the weighted sum of latest N input data samples

$$y[n] = \sum_{i=0}^{N} A[i]x[n-i] \tag{5.1}$$

The weights A[i] in the equation 1 are the filter coefficients. The number of taps (N) and the coefficient values are derived so as to satisfy the desired filter response in terms of pass-band ripple and stop-band attenuation. FIR filters with symmetric coefficients A[i] = A[N-1-i] have a linear phase response (0 group delay) and are hence an ideal choice for applications requiring minimal phase distortion. A general FIR filter architecture is shown in figure 5.1.

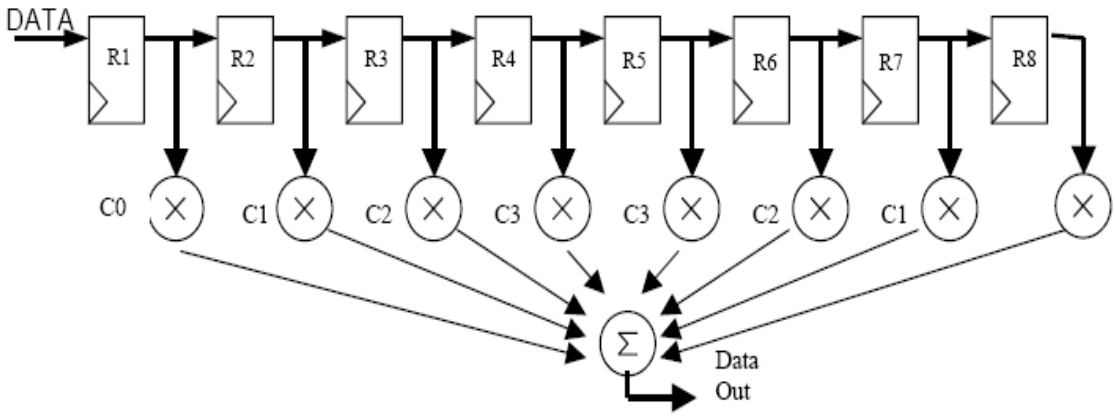


Figure 5. 1: Tap delay line filter or transversal FIR filter

The input is shifted through 8 registers (taps). Each output stage of a particular register is multiplied by a known coefficient. The resulting outputs of the multipliers are then summed to create the filter output. Note that the coefficients are symmetric about the centre taps. This is true for a linear phase response FIR filter as stated above. So this will allow us to "fold" (folding is an operation done by replacing the independent variable n by –n. it is the reflection of the signal about the time origin n=0. This is done while convolving the signal with another). The filter in half as shown in figure 5.2 thus reduces the amount of multipliers to half.

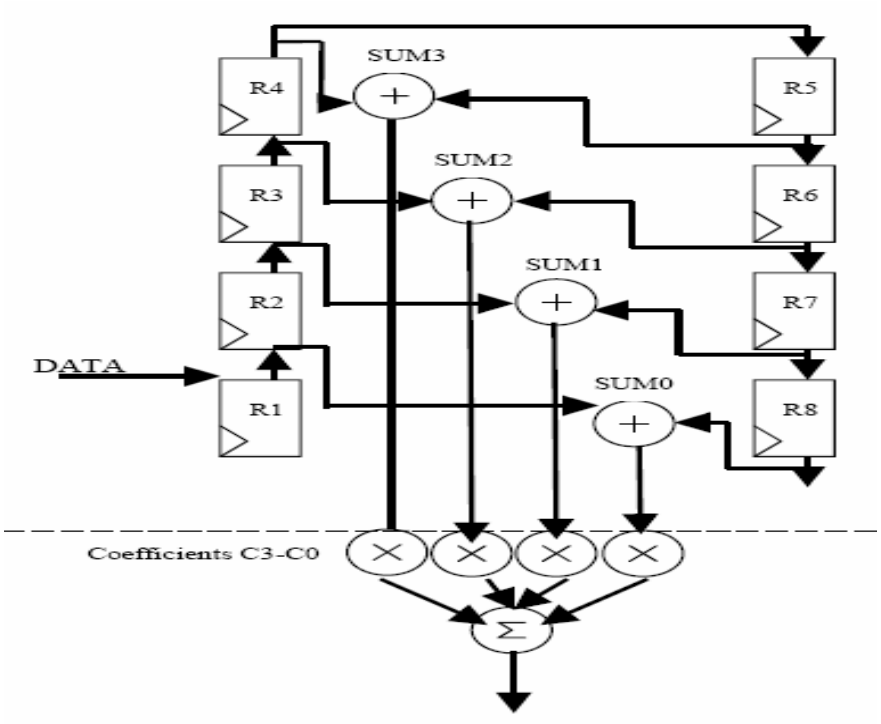


Figure 5. 2: Folded direct form FIR filter architecture

5.3 Hearing Aid structure

The test circuit taken for various clocking scheme applied to the front end of a noise suppression circuit combines signals from two microphones. Such circuits find applications in hearing aids to augment speech intelligibility. A block diagram is shown in figure 5.3. The two FIR filters (see inset in figure 5.2) are implemented in fixed-point arithmetic (quantization details and coefficients are also shown in figure 5.3) and make use of standard low-power techniques, such as hybrid number representation, sign-magnitude (SM) in the multiplication, 2's complement (2'sC) for the rest, pre-addition, and register files (as opposed to shift registers where data samples turn in circles [93, 97, 104, 117-118].

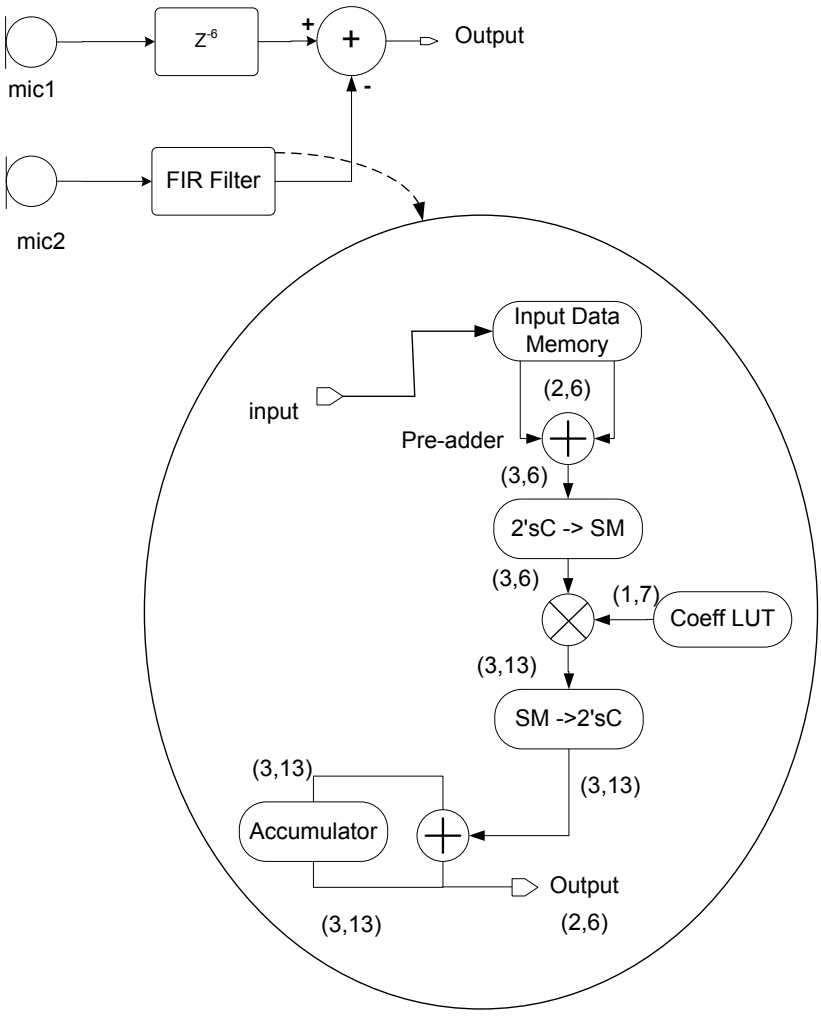


Figure 5. 3: Block diagram of front end hearing aid

5.4 Clocking Strategy

Proper timing and clock system design are one of the most critical components in digital system. In digital system the clock designates the exact moment when the signal is to change as well as its final value is to be captured, when the logic is active or inactive. Finally, all the logic operations have to finish before the tick of the clock and the final values of the signals are being captured at the tick of the clock. Therefore, the clock provides the time reference point which determines the movement of data in the digital system [94-96, 98-102, 108-116].

5.4.1 Clock Signals

Clocks are defined as pulsed, synchronizing signals that provide the time reference for the movement of data in the synchronous digital system. The clocking in a digital system can be either single phase, multi-phase (usually two-phase) or edge-triggered, as shown in figure 5.4. The dark rectangles in the figure represent the interval during which the bi-stable element samples its data input. Figure shows the possible types of clocking techniques and corresponding general finite-state machine structures:

a) Single-phase clocking and single-phase latch machine, figure 5.4 (a)
b) Edge-triggered clocking and flip-flop machine, figure 5.4 (b)
c) Two-phase clocking and two-phase latch machine with single latch figure 5.4
d) Two-phase clocking and two-phase latch machine with double latch figure 5.4 (d)

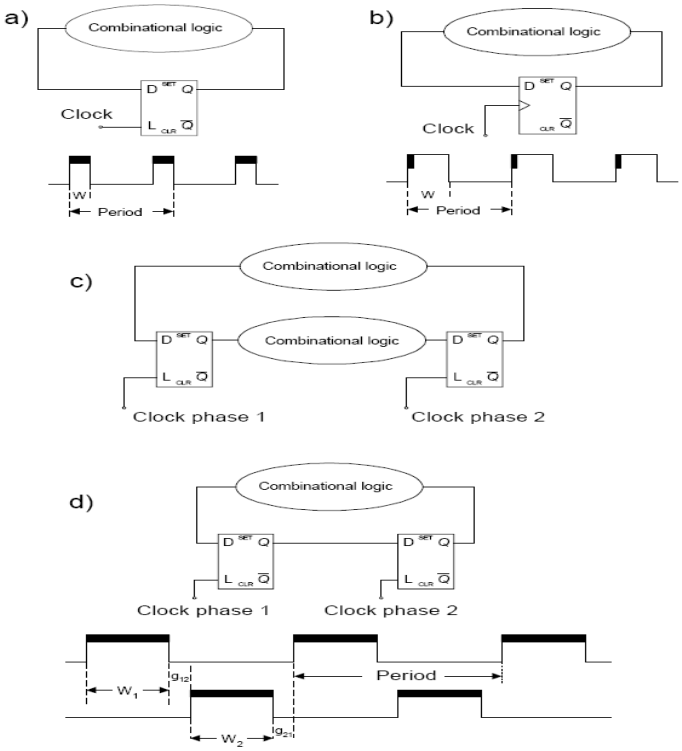


Figure 5. 4: System clocking waveforms and general finite-state machines structures

In figure 5.4 (c) and (d), W_j is the pulse width of the phase j and g_{ij} is the inter-phase gap from phase i to phase j; if $g_{ij} > 0 \Rightarrow$ two-phase, non-overlapping, if $g_{ij} < 0 \Rightarrow$ two-phase, overlapping clocking scheme. The multi-phase design typically extends to three, but not more than four non-overlapping phases. Two non-overlapping clocks provide most reliable and robust clocking system that fits well into the design for test-ability methodology.

5.4.1.1 Clock Distribution

The two most important timing parameters affecting the clock signal are: (a) Clock Skew and (b) Clock Jitter. Clock Skew is a spatial variation of the clock signal as distributed through the system shown in figure 5.5. It is caused by the various RC characteristics of the clock [94-96, 98-102, 108-116] paths to the various points in the system, as well as different loading of the clock signal at different points on the chip.

Clock Jitter is a temporal variation of the clock signal with regard to the reference transition (reference edge) of the clock signal as illustrated in Figure 5.5. Clock jitter represents edge-to-edge variation of the clock signal in time. As such clock jitter can also be classified as: long-term jitter and edge-to-edge clock jitter, which defines clock signal variation between two consecutive clock edges. In the course of high speed logic design we are more concerned about edge-to-edge clock jitter because it is

this phenomenon that affects the time available to the logic. Typically the clock signal has to be distributed over several hundreds of thousands of the clocked storage elements (also known as flip-flops and latches). In addition clock also has to support various levels of amplification (buffering). As a consequence, the clock system by itself can therefore; the clock signal has the largest fan-out at any node in the design that requires up to 40-50% of the power of the entire chip. Also it must be assured that every clocked storage element receives the clock signal precisely at the same moment in time.

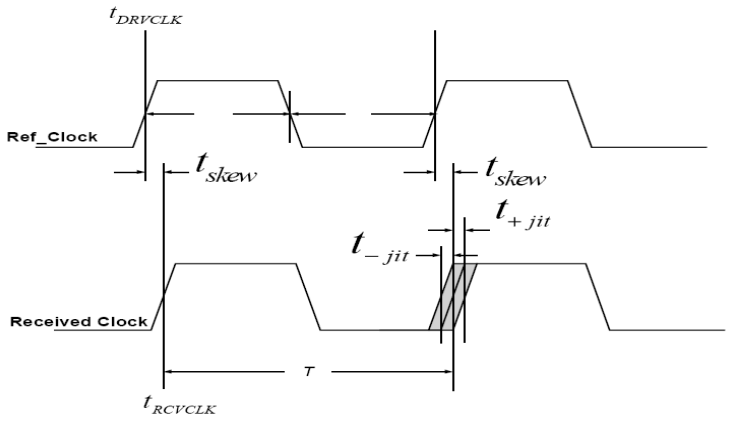


Figure 5. 5: Clock parameters: period, width, clock skew and clock jitter

5.4.1.2 Bi-Stable Elements

In order to define the clocking of the system properly, the nature and behaviour of the bi-stable elements often referred as a “latch” or “Flip-Flop” needs to be specified precisely. It is quite common to find both terms “Flip-Flop” and a “Latch” to be used indiscriminately for the bi-stable element used in the synchronous systems. In a synchronous system, operations and data sequences take place with a fixed and predetermined time relationship. The timing of computations are controlled by flip-flops and latches together with a global clock. Flip-flops and latches are clocked storage elements, which store values applied to their inputs. They are classified according to their behaviour during the clock phases. A latch is level sensitive. It is transparent and propagates its input to the output during one clock phase (clock low or high), while holding its value during the other clock phase. A flip-flop is edge triggered. It captures its input and propagates it to the output at a clock edge (rising or falling), while keeps the output constant at any other time. The design of these clocked storage elements are highly depended on the clocking strategy and circuit topology. Since we are interested in power reduction of the latch, flip-flop, clocking strategy plays a very im-

portant role. Hence we discuss these circuits before we take up an appropriate clocking strategy.

Latch

It is a device capable of storing the value of the input D in conjunction with the clock C and providing it at its output Q. The latch has a following relationship between the input D and the clock C: While C=0 the output Q remains constant regardless of the value of D (the latch is "opaque"). While C=1 the output Q=D and it reflects all the changes of D (the latch is *"transparent"*). Often we describe such behaviour of a latch as *"level sensitive"* (the behaviour of the latch is dependent on the value of the clock - not the changes). Behaviour of an ideal Latch is illustrated in figure 5.6.

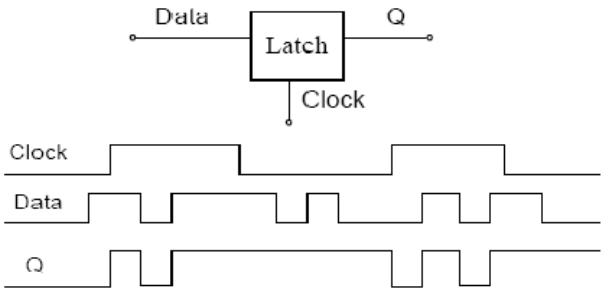


Figure 5. 6: Signal relationship of an ideal latch.

Flip-Flop

It is defined as a bi-stable memory element with the same inputs and outputs as a "latch", in figure 5.7. However, the output Q responds to the changes of D only at the moment the clock is making transitions. We define this as being *"edge triggered"*. The internal mechanisms of the "flip-flop" and that of a "latch" are entirely different. We further define a "Flip-Flop" as a *"leading edge triggered"* if the output Q assumes a value of the input D as a result of the transition of the clock C from 0-to-1. Conversely in a *"negative edge triggered"* Flip-Flop the output Q assumes a value of the input D as a result of the transition of the clock from 1-to-0. It is also possible to build *a "double-edge triggered"* flip-flop that responds to both: *leading* and *trailing* edge of the clock.

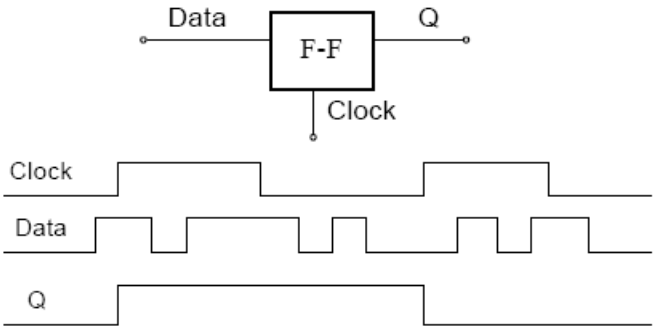


Figure 5. 7: Signal relationship of an ideal leading-edge triggered flip-flop

5.4.2 Single Edge Triggered Flip Flop & Dual Edge Triggered Flip Flop

When the two latches are connected in series as shown in figure 5.8 then the flip-flop results is called single edge triggered flip flop. The data are loaded at only one clock edge, either rising or falling.

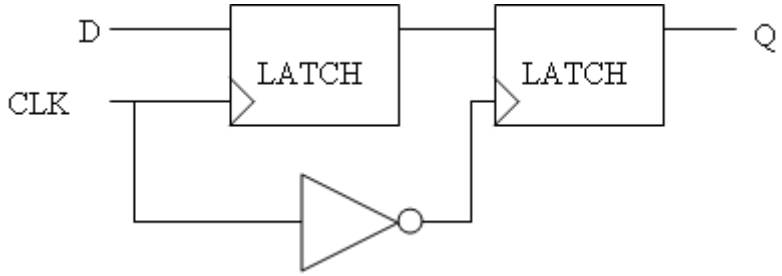


Figure 5. 8: Single edge triggered flip-flop

And when the two latches are connected in parallel as shown in figure 5.9 the flip-flop results is called dual edge triggered flip flop. The data are loaded at both rising and falling clock edge. A dual edge triggered flip-flop requires slightly more transistors to implement.

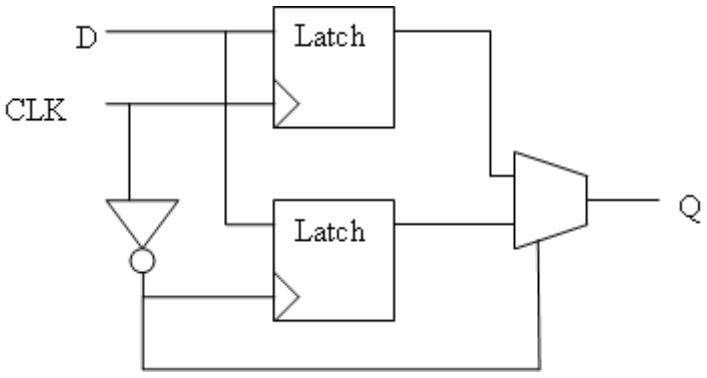


Figure 5. 9: Dual edge triggered flip-flop

5.4.2.1 Clock Gating

Low-power techniques are essential in modern VLSI design due to the continuous increase of clock frequency and chip complexity [119]. In particular, the clock system, composed by flip-flops and clock distribution network, is one of the most power consuming subsystems in a VLSI circuit. The three major components of power consumption:

1. Power consumed by combinational logic whose values are changing on each clock edge.
2. Power consumed by flip-flops (this has non-zero value even if the inputs to the flip-flops, and therefore, the internal state of the flip-flops, is not changing)
3. The power consumed by the clock buffer tree in the design.

As a consequence many techniques have been proposed to reduce clock system power dissipation. Disabling the clock signal (clock gating) in inactive portions of the chip is a useful approach for power dissipation reduction. Clock gating can be applied to different hierarchical levels. It is possible to disable the clock signal that drives a big functional unit reducing power dissipation on both its internal nodes and its clock line. Automatic clock gating is supported by modern EDA tools. They identify the circuits where clock gating can be inserted.

Clock gating works by identifying groups of flip-flops which share a common enable control signal. Traditional methodologies use this enable term to control the select on a multiplexer connected to the D port of the flip-flop or to control the clock enable pin on a flip-flop with clock enable capabilities.

Clock gating principle involves in freezing the clock to the portions of a design that are ideal or are not performing useful computations. This technique is successful and widely used for power reduction. The figure below describes the different concepts of clock gating. Independent blocks when clock gated, it greatly affects power savings because gating larger blocks achieves in higher power savings in ‘off’ clock cycle. There are various types of clock gating [119] like:

1. module level clock gating,
2. register level of clock gating
3. Cell level clock gating.

- **Module level clock gating**

This method shuts of the entire block or module in the design. This method saves large amount of power when entire block or module is not functioning. In cases like, when a block is used for a specific mode of operation (example of receiver and transmitter part of transceivers). These techniques are limited and only identified the RTL designer.

- **Register level of clock gating**

In this method the clock to a single set of register or a set of register is gated. Synchronous load enabled registers are usually implemented using a clocked D flip-flop and encircling the multiplexer as shown in figure with d flip-flop figure 5.10(a), which is clocked every cycle. Clocked gate of figure a is shown in figure 5.10(b), the clocked version of the above two circuits are shown in figure 5.10(c) and figure 5.10(d). In clocked gate version, register does not get the clock signal until no new data is loaded. Removing the multiplexer also saves power but gating a single register consumes plenty of power. In order to save power at least a single clock gating circuit should be used.

Though power saving in case of register level clock gating than in module level gating, register level clock gating gives more opportunities to shut off clocks. It can be easily automated for massively clocked cells than module based clock gating.

- **Cell level clock gating**

In cell level clock gating the cell designer inserts the clock gating. Let us take a register bank that will receive the clock only when register is loaded with new data. Similarly a memory block may be clocked once during active clock cycle and this is an easier method to implement. But this may not be useful because of the area overhead which will limit the power saving as all the register should be predesigned with clock gating. Also clock gating can not be shred across many registers. This method of implementation generally not used.

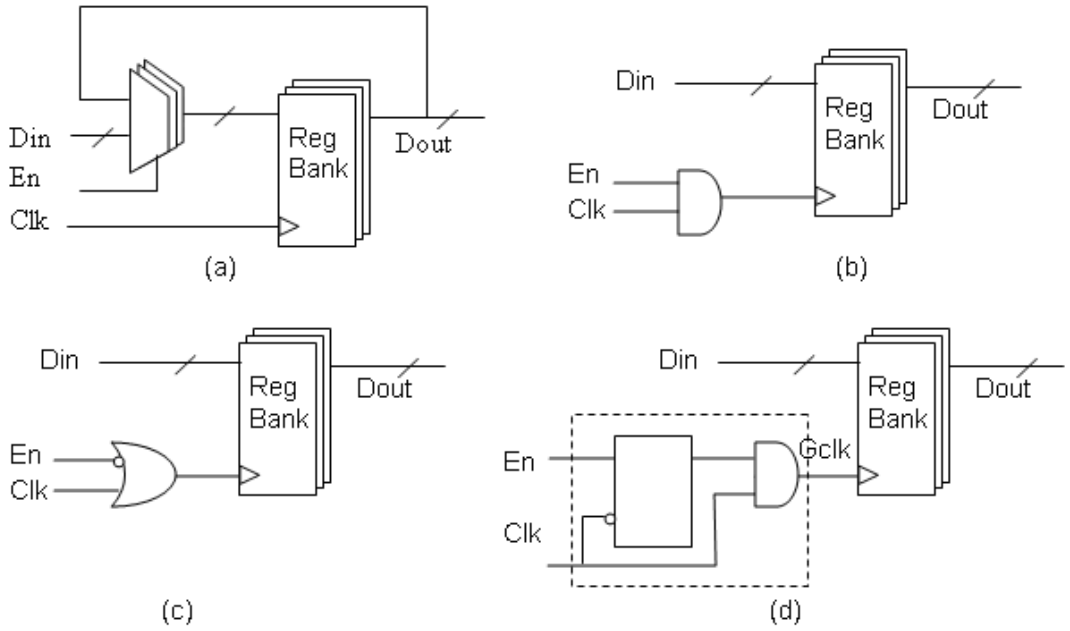


Figure 5. 10: Clock gating at various levels

In simple clock gating as shown in figure 5.10(b), glitches that occurs on the enable signal at high edge of clock can propagate to the clock pin of the register,. This glitch can be prevented by giving appropriate setup and hold constraint on the enable signal at the input of the AND gate. Any spurious change in run time may cause wrong values to be latched into the gated register. So it is better to use an OR gate as shown in figure 5.10(c). This gate holds logic '1' at output when enable signal is high. During running time spurious glitches may cause wrong data to be loaded into flip-fop but the final value at the rising edge of clock will be correctly available.

Another method to avoid this by adding a level sensitive active low latch after enable signal as shown in figure 5.10(d). This freezes the latch output at the rising edge of the clock and ensures that new enable signal to be stable at the AND gate input when clock is high, or falling edge flip-flop can be used instead of latch that gives a clean signal to the AND gate which makes enable signal to be stable before falling edge of clock.

5.4.2.2 Clocking Schemes

Single-edge-triggered (SET) one-phase clocking is certainly the most common choice. There are five good reasons for this

1. Automata theory maps to SET operation directly

2. It is supported by all common design tools
3. It requires the routing of only one clock tree
4. The basic bi-stable (SET-FF) is relatively simple,
5. A scan chain is easy to implement for testing

SET clocking has important drawbacks in terms of energy efficiency, however. This clocking scheme has found to have following three drawbacks:

1. Superfluous switching whenever a register is disabled,
2. The presence of an inactive clock edge,
3. The strong vulnerability to clock skew.

The first limitation is commonly addressed with clock gating. Also an idea of dual edge triggered clocking is proposed to overcome the second limitation. Also with clock period reduced to 200 ps nowadays, the importance of clock uncertainties will increase, and complex multiple-phase clocking will become impractical due to increasingly large timing uncertainties and power consumption.

In order to save on clocking power, dual-edge triggered (DET) clocking strategy uses DET storage elements (DETSE) that capture the value of the input after both rising and falling clock transitions. Otherwise, DETSE is non-transparent, i.e., it holds the captured value at the output. Thus, the DET clocking strategy provides a one-time solution to the frequency scaling by retaining the data throughput of single-edge triggered (SET) clocking at halved clock frequency. However, in order to fully exploit the power savings in the clock distribution network, this approach must ensure that the delay and energy consumption of DETSE must be comparable to those of SET storage elements (SETSE). Furthermore, use of both clock edges to synchronize the operation makes timing sensitive to clock duty cycle and increases clock uncertainties generated by the clock distribution system.

5.4.3 Power Reduction Using Dual Edge Triggered Clocking Strategy

As discussed earlier, clock related power is one of the most significant components of the dynamic power consumption. The total clock [94-96, 98-102, 108-116] related power dissipation in synchronous VLSI circuits is further divided into three major components. The total power dissipation of the clock network depends on both the clock frequency and the data rate, and can be computed based on equation

$$P_{CK} = V_{dd}^2[f_{CK}(C_{CK} + C_{ff,CK}) + f_D C_{ff,D}] \quad (5.2)$$

Where f_{CK} is the clock frequency.

f_D is the average data ratse.

C_{CK} is the total capacitance seen by the clock network.

$C_{ff,CK}$ is the capacitance of the clock path seen by the flip-flop.

$C_{ff,D}$ is the capacitance of the data path seen by the flip-flop.

From equation 5.2 it is obvious that the clock power can be reduced if any of the parameters on the right hand side of the equation is reduced. The reduction of Vdd is already the trend of contemporary design, and it has the strongest impact on the P_{CK} expression. By reducing the overall capacitance of the clock network, C_{CK}, the power dissipation may also be reduced. For instance, the capacitance can be reduced by proper design of clock drivers and buffers. Similarly, by reducing the capacitance inside a flip-flop, $C_{ff,CK}$ and $C_{ff,D}$, power may also be reduced. Furthermore, the clock power dissipation is linearly proportional to the clock frequency. Although the clock frequency is determined by the system specifications, it can be reduced with the use of dual edge triggered flip-flops (DETFFs). As its name implied, DETFF responds to both rising and falling clock edges. Hence, it can reduce the clock frequency by half while keeping the same data throughput. As a result, power consumption of the clock distribution network is reduced, making DETFFs desirable for low power applications. Even for high performance applications, the usage of DETFFs offers certain benefits. Since the clock speed is reduced by a factor of two, one does not need to propagate a relatively high speed clock signal.

5.4.3.1 Proposed Glitch Reducing Clocking Technique

Glitches are responsible for a significant proportion of overall power dissipation in digital signal processing circuits. Activity-reduction techniques that involve an optimized clocking strategy have been applied to a front-end block in a DSP- adaptive directional microphone for hearing aids.

As a consequence from the third limitation of SET clocking scheme, its operation necessitates careful balancing of clock distribution delays and fast clock ramps. These two requirements are typically met with multiple clock buffers of generous size and power. Level-sensitive two phase clocking does away with the need for fast

ramps as the non-overlap phases provide welcome skew margins. Yet the extra load of a second clock tree reduces the benefits. Also the clock distribution never absorbs more than 30%-40% of a chip's overall power dissipation. Even if clock power could be cut in half using DET clocking scheme, this would translate into a mere 15% to 20% overall savings. So a symmetric Level sensitive two phase clocking is undertaken to mitigate spurious switching activities in the data-path too.

This achievement has been made possible by combining two novel techniques:

1. A multi-stage clock gating
2. A symmetric two-phase level sensitive clocking with glitch-aware re-distribution of data-path registers.

The starting point is a symmetric two-phase level sensitive clocking which is again enhanced by two features:

1. Multi-Stage clock gating
2. Stop-Glitch latch barriers

5.4.3.2 Multi-Stage Clock Gating

Multi-stage clock gating implies that multiple clock gates are cascaded hierarchically to minimize the useless toggling of heavily loaded nodes. While this technique is already supported in well-known synthesis tools such as Synopsys for SET clocking, it has never been addressed in the context of two-phase clocking. Non-overlap phases provide ample skew margins and make multi-stage clock gating particularly easy to combine with level-sensitive two-phase clocking. This contrasts favourably with edge-triggered designs, where meagre hold margins make it difficult to avoid timing violations.

This multi stage clock gating has been applied to the FIR filters. Accessing a memory in a circular fashion typically involves a counter and an address decoder. If combined with level sensitive two phase clocking and clock gating, a multitude of AND gates result as shown in figure 5.11 which altogether, are responsible for a heavy capacitive load on the clock net. By re-arranging the decoder in a hierarchical fashion, capacitive load can be significantly lowered. The re-arranged decoder is shown in figure 5.12.

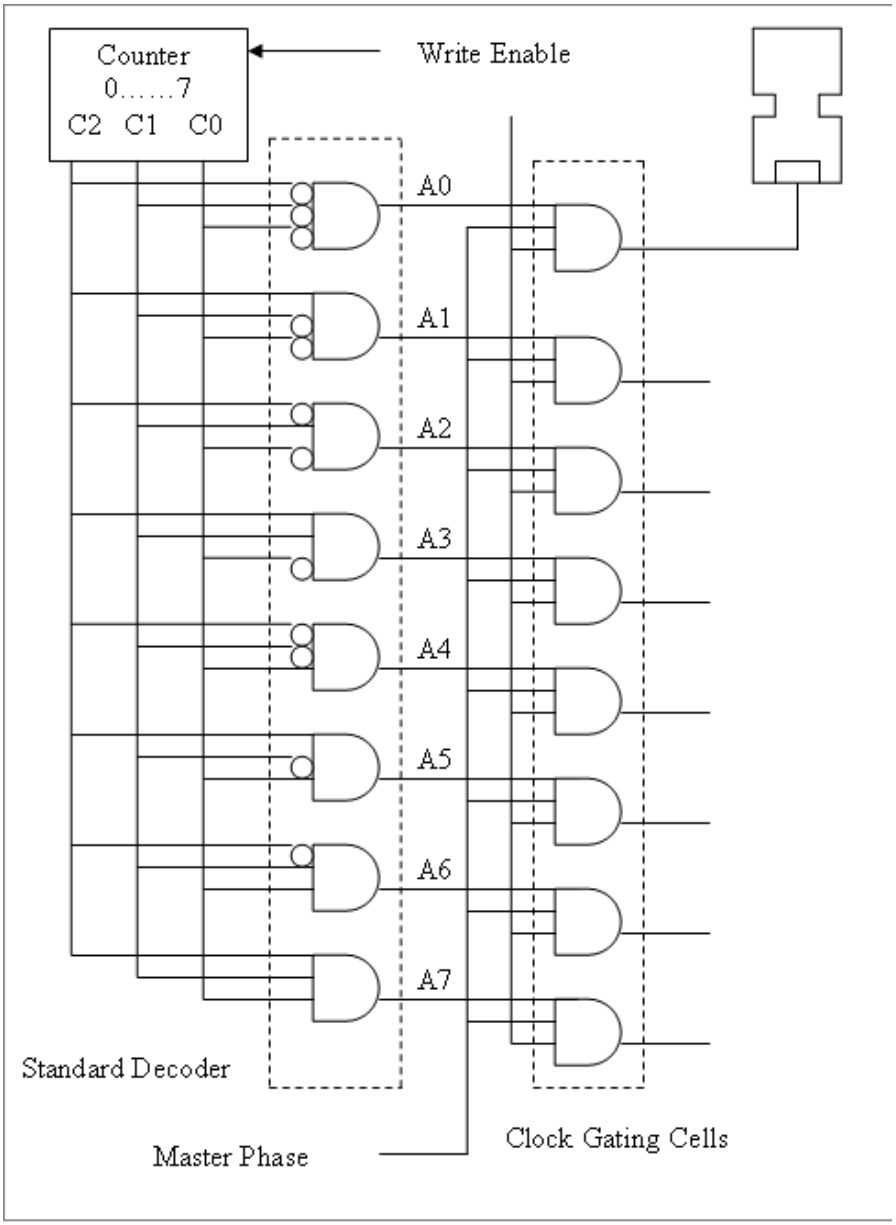


Figure 5. 11: Traditional clock gating applied to a register file

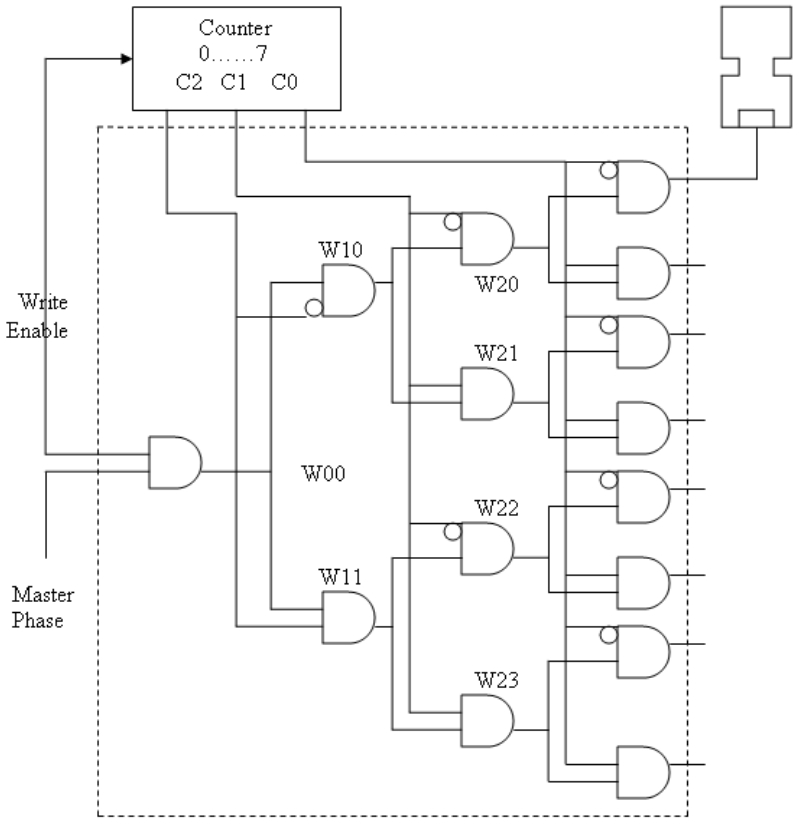


Figure 5. 12: Multistage clock gating decoder

The relevant quantity is the activity capacitance product summed up over all circuit nodes driven from the clock. This sum of products will be referred to as "effective capacitance". For a traditional address decoder the effective capacitance can be calculated from node switching activities (α). Given a memory of size n (a power of two), and defining k as $\log_2$ n, the switching activities of the counter outputs c_0……c_{k-1} are:

$$\alpha_{C(0)} = 1 \quad \alpha_{C(1)} = 1/2 \quad \ldots\ldots\ldots\ldots\ldots \quad \alpha_{C(k-1)} = 1/2^{k-1} \qquad (5.3)$$

The signals q_0……q_{n-1} toggle only twice during a whole writing cycle, therefore

$$\alpha_{qj} = 2/n = 1/2^{k-1} \qquad j = 0, \ldots\ldots\ldots\ldots, n-1 \qquad (5.4$$

The input gate capacitance is C_0 and the switching activity of the master phase a,m is two by definition. The effective capacitance of the traditional decoder (C_1) can be expressed through proper combination of equation .5.3 and Eq.5.4so that

$$C_1 = \sum_{J=0}^{K-1} \alpha_{cj}.n.C_0 + \sum_{J=0}^{N-1} \alpha_{QJ} C_0 + \alpha_M .n.C_0 = 4nC_0 \qquad (5.5)$$

After re-arranging the decoder in a hierarchical way, the switching activity of the nodes w_{ij} can be expressed as:

$$\alpha_{w00} = 2,. \ . \ . \ . \ . \alpha_{wij} = 1/2^{i-1} \tag{5.6}$$

From equation (5.3) and (5.6), let express the effective capacitance of a multi-stage decoder (C_2)

$$\begin{aligned} C_2 &= (2\alpha_{c(k-1)} + \ldots + n/2\,\alpha_{c1} + n\alpha_{C(0)}C_0 + \sum_{i=0}^{k-1} 2\sum_{j=0}^{2^i-1}\alpha_{wij}C_0 + \alpha_m C_0 \\ &= nC_0\sum_{i=0}^{k-1}\left(1/4\right)^i + 4kC_0 + 2C_) = (4(n^2-1)/3n + 4\log_2 n + 2)C_0 \end{aligned} \tag{5.7}$$

The saving in dynamic energy obtained from multi-stage clock gating can be expressed as follows equation (5.7)

$$\Delta E/E = C_1 - C_2/C_1 \xrightarrow{n\to\infty} 2/3 \tag{5.8}$$

Equation (5.8) suggests that up to 67% of dynamic energy can be saved when the circular buffers those grow very large. Moreover, the multi-stage clock gating is n times more efficient than the flat clock gating scheme when no memory access occurs (write enable remains passive). This happens because the master clock is locked out at the root of the address decoder tree (figure 5.11), avoiding any unproductive switching activity. Thus, large buffers with relatively few write accesses, such as those found in high-order FIR filters, benefits the most from hierarchical clock gating.

Stop-Glitch Latch Barriers

Symmetric two-phase level-sensitive clocking brings about a degree of freedom unavailable with the other clocking disciplines, namely the exact location of all slave-triggered latch banks. Relocating them can have a large impact on glitch activities and, hence, on the overall energy efficiency. The most advantageous place is just in front of the most energy-hungry circuit blocks in a data-path. The controller may require some changes too, but latency will remain the same.

5.5 Low Power Latch

In order to exploit the maximum benefits from the clocking techniques introduced before, a low-power latch circuit has been designed. As discussed earlier two latches in parallel can implement a double edge triggered flip-flop. In line with the on-going trend towards low-power, low-voltage operation the circuit of D-Latch shown is original low power circuit [120-124]. But it suffers from sub-threshold currents if there is a voltage difference across the feedback PMOS pass transistor. Furthermore, the implementation of a weak feedback inverter in figure 5.13 will not be beneficial to power saving as it will worsen the low level imbalance across the feedback PMOS pass transistor.

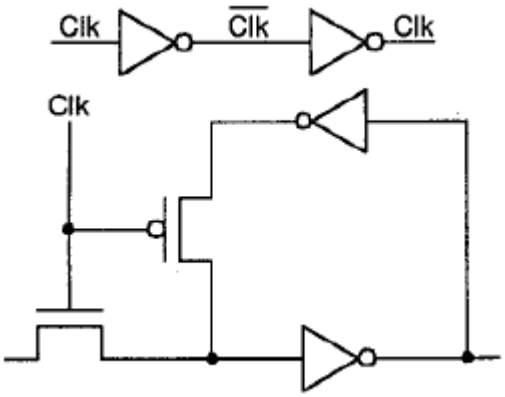


Figure 5. 13: Block diagram of original low power D-latch

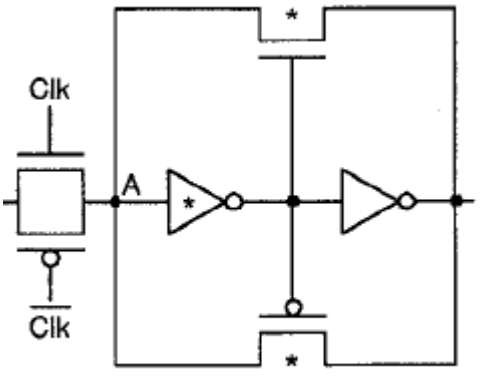


Figure 5. 14: Block diagram low power and low voltage D-latch

Instead, figure 5.14 shows a circuit with a total of eight transistors. The main advantage of this configuration is that weak inverters and pass transistors marked with '*' are incorporated to drastically reduce the standby power dissipation. The weak in-

verter can sufficiently drive the two weak pass transistors and another regular-sized inverter. Similarly, the weak pass transistors only need to maintain the voltage at node A when the transmission gate is switched off. Thus, the storage node A can withstand a noisy environment and the D-latch is fully static. In addition, both the pass transistors have equal voltage at their sources and drains at all times, eliminating the problem of sub-threshold current leakage. Only the transmission gate and the output inverter need the normal size to maintain the driving capability. Hence, this novel D-latch has very low standby power dissipation and is suitable for very-low-voltage applications.

Applying this idea a simple 10 transistors low power latch circuit is designed as shown in figure 5.16. This circuit of 10 transistors when used against the 21 transistors based circuit out of the technology library occupying roughly one half of the area.

The circuit operation is as follows: From the figure 5.15 assuming that an input of V_{dd}-V_{tn} is applied during previous cycle when clock (clk)=1, which causes circuit output to go low. The effect of new input (low) turns off the n-transistor (M6) and M3 is turned on. This causes input node voltage at A to decrease and as soon as A falls below V_{dd}- $|V_{tp}|$, M_4 is turned on. When voltage at node B falls rises above $|V_{tp}|$, M_1 becomes off, which allows voltage at node A to fall faster. This switching action is regenerative type so node voltage at B is pulled to $|V_{tp}|$ above ground.

Assuming an input low is applied in next transparent cycle of clock, causing output to be at high. When input high is applied (i.e. $V_{dd} – V_{tn}$) is transferred from input which turns on the transistor M6 and M3 is turned off thereby M8 is pulled to off. When clock signal becomes high, voltage at node falls below V_{dd}-$|V_{tp}|$, M1 is turned on, thereby node voltage at B rises.

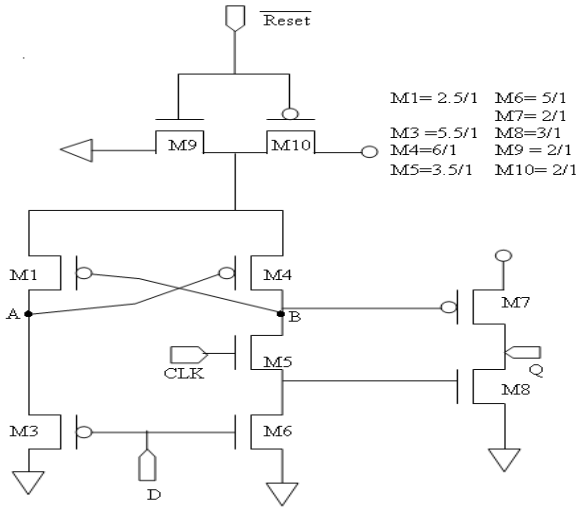


Figure 5. 15: low power latch circuits using 9 transistors

As voltage at B rises, M4 turns off, causing voltage at A to fall at a faster rate. Due to this M7 turns on. Therefore, logic high is observed at output. From this analysis, we can see that some power is dissipated in the latch due to fact that M3 is no completely turned off.

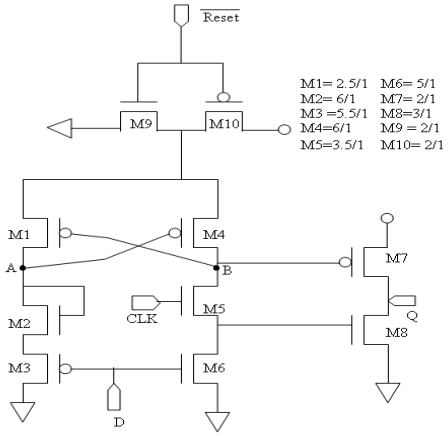


Figure 5. 16: Low power latch circuits using 10 transistors

M1 and M3 must be properly scaled to control the current flows through M3, as the only load that is driven by node A are drain to source capacitance of M2 and M3 and gate capacitance of M4 so that switching activity can be minimized. In one trade off exists that more current flow during switching of node B as there exists a feedback action of M1 and M6.

Due to this there exists static power dissipation. So to minimize the a transistor M2 is connected as shown in figure 5.16 which reduces the potential at drain of M3 to V_{dd}-V_{tn} for an input high. The gate of M3 is also at V_{dd}- V_{tn}. From simulation, we observed that power consumption with M2 is less than without M2. The simulation results for figures 5.15 and 5.16 are shown in figures 5.17, 5.18 respectively.

From figure 5.17 and 5.18 it can be seen that latch working according to our expectations. The power consumed by the latch without the M2 is about 21 μW and with M2 is around 22 μW in which the power supply is kept at 3.3V and clock frequency at 250 MHz. For the simulation purpose we have taken 350 nm model file.

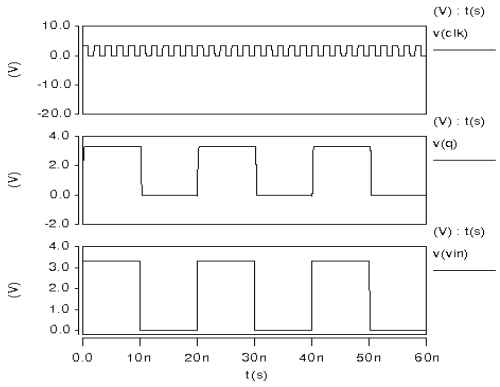


Figure 5. 17 Result of dff with 350nm tech at 3.3V with power loss of 21μW at 250 MHz

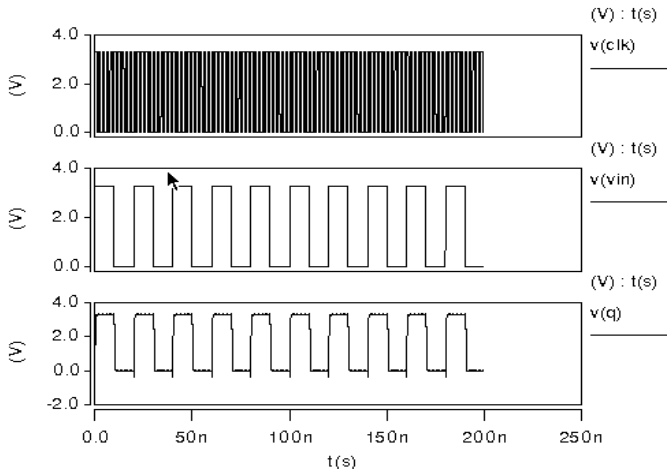


Figure 5. 18: Result of dff with 350nm tech with reset logic tech 3.3V with power loss of 22μW at 250 MHz

The figure 5.19 shows the simulation output of figure 5.15 with reset logic taking 180 nm technology at 3.3 V power supply and clock frequency of 250 MHz. this circuit consumes a power of 190 pW.

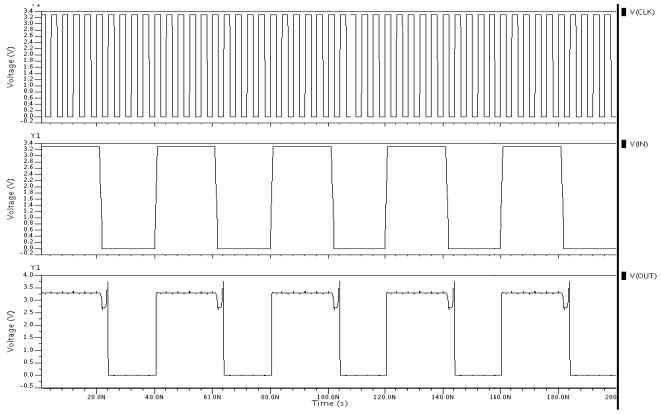


Figure 5. 19: Simulation result of the circuit 5.16 with 180 nm technology.

The layout structure of figure 5.16 with 180 nm technology is shown in figure 5.20 and its simulation result is shown in figure 5.21. From the figure 5.20 it can be seen that the output matches our result in figure 5.18 and there exists small amount of spike due to the existence of the capacitors and resistors in the layout extraction.

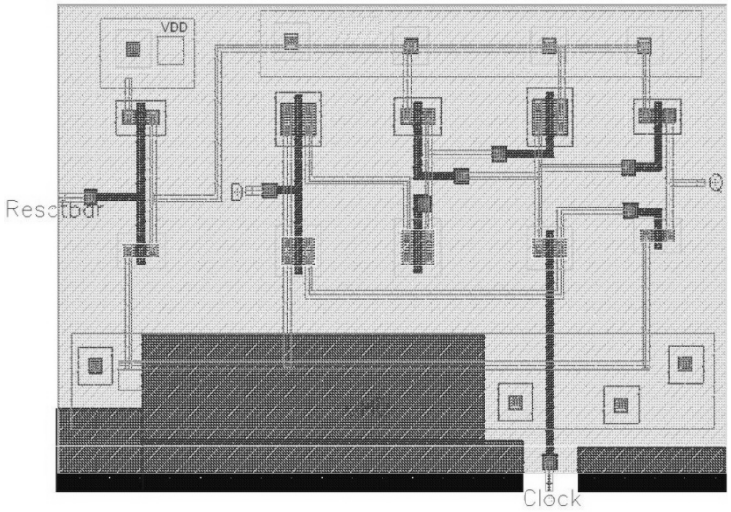


Figure 5. 20: Proposed low power latch layout with reset.

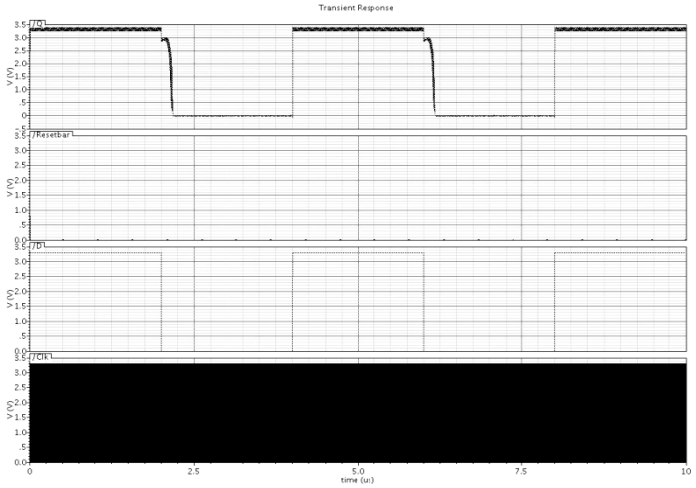


Figure 5. 21: Post layout simulation of the figure 5.16

Figure 5.20 show the layout of figure 5.17 and figure 5.21 shows the post layout simulation. Observing the result in figure 5.19 and 5.21, we conclude that our proposed latch works satisfactorily.

5.6 MAC based FIR Filter Structures and Simulation and **Conclusion**

Using clocking strategy discussed in previous sections the new MAC based filter structures are given in figure below with SET based latch. The MAC based FIR filter given figure 5.22 uses traditional clocking and the MAC based FIR filter in figure 5.23 uses DET based clocking strategy for glitch reduction.

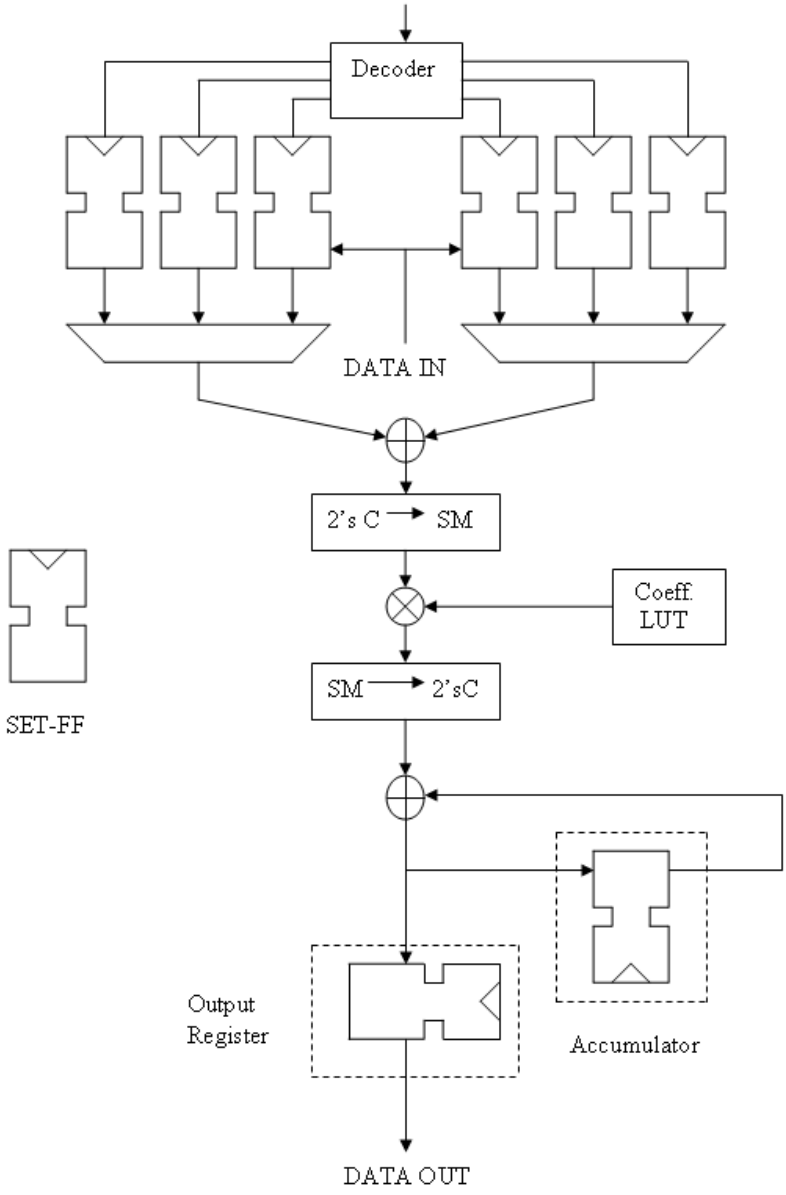


Figure 5. 22: Block diagram of FIR filter with traditional clocking

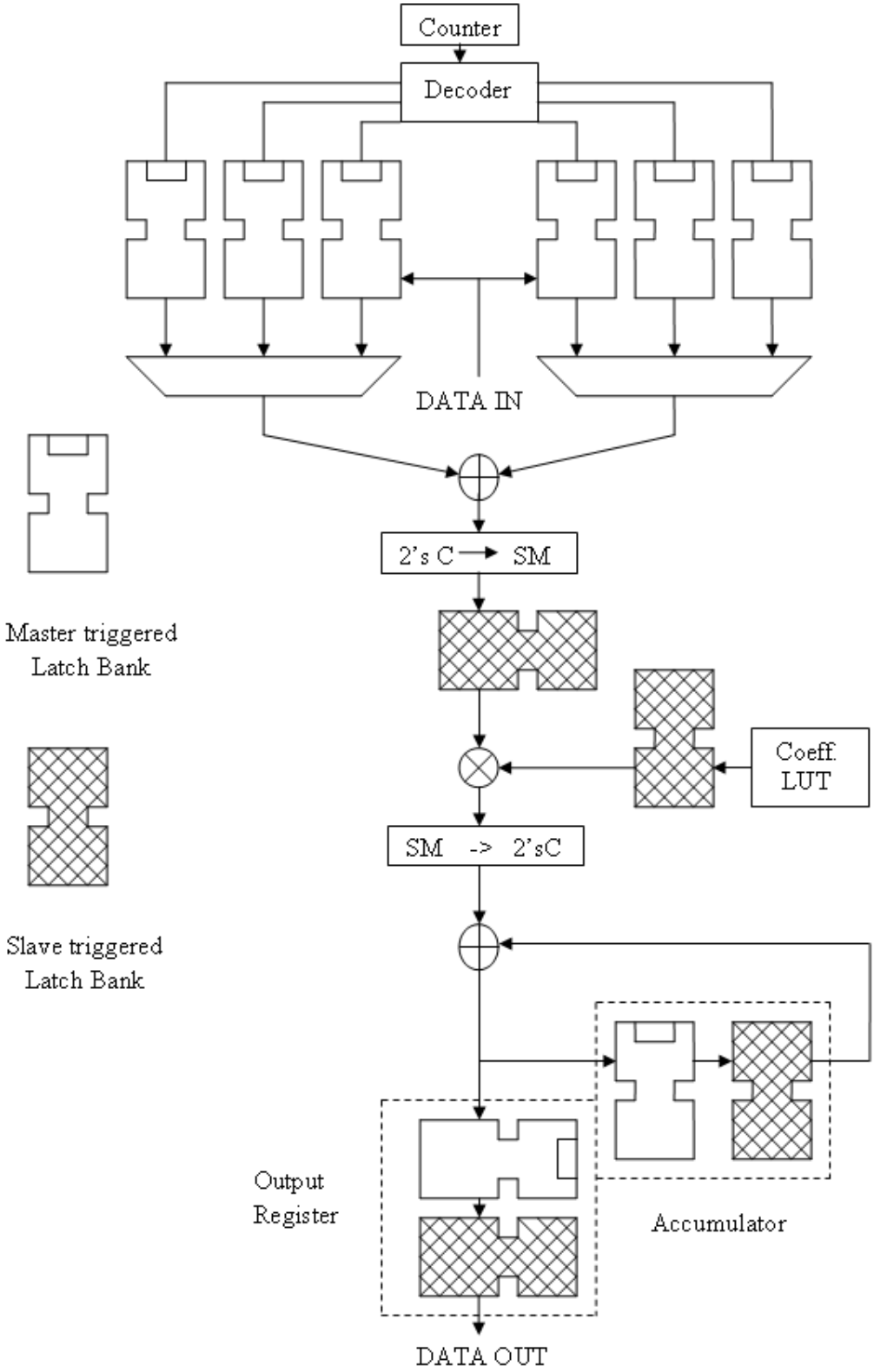


Figure 5. 23:block diagram of the FIR filter with dual edge clock gating strategy

5.6.1 Simulation Results

5.6.1.1 Simulation Result of an ALP for a 6-Tap FIR Filter

A simple program for a 6-Tap FIR Filter is written using the instructions of the processors. Basically the program in assembly language for FIR filter consist of load instructions to get the input data and coefficients, then the basic operation involved is multiply and accumulate only. The multiplier designed in the processor takes 6 clock cycles to multiply two 4-bit numbers. The simulation results are shown in figure 5.24. The simulation results shows data in hexadecimal (8 bits) which is the op code value of the instructions used the input and the filter coefficient value. The path out is the output port shows the decimal value (8-bits) of the output of the FIR filter. Also the ALU flag shows the status of various flags as per the execution of different instructions.

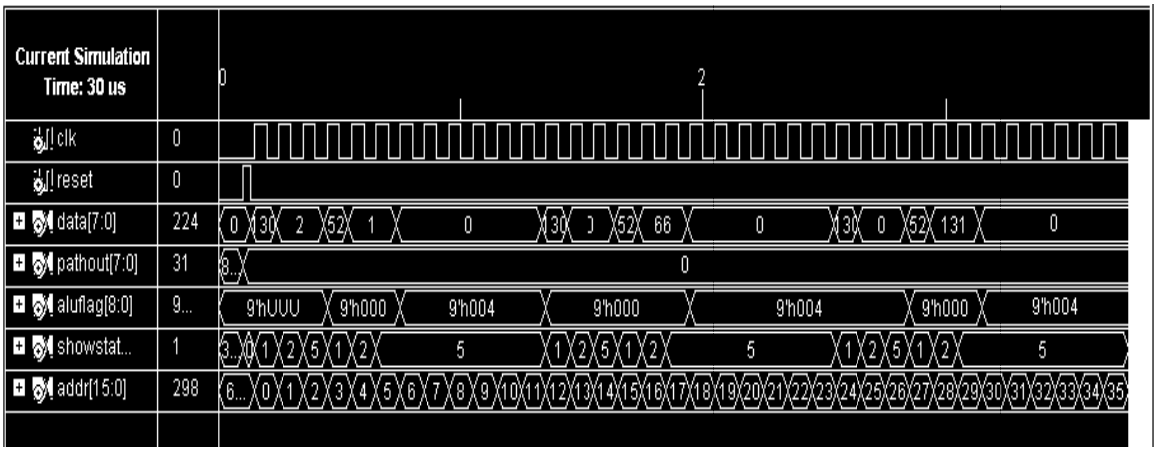


Figure 5. 24: Simulation output of a 6-Tap FIR filter program

5.6.1.2 Simulation Result of FIR Filter with Single edge Triggered Clocking.

Figure 5.25 shows the simulation output of the FIR Filter Block Diagram shown in figure 5.22. It uses the single edge triggered clocking strategy. It takes less time to produce the output in comparison to the proposed clocking strategy. In the simulation output the input port datain takes the input of FIR filter. Another input port coeffLUT accepts the FIR coefficients. The output port dataout gives the output of FIR filter. All the values of these ports are displayed in decimal.

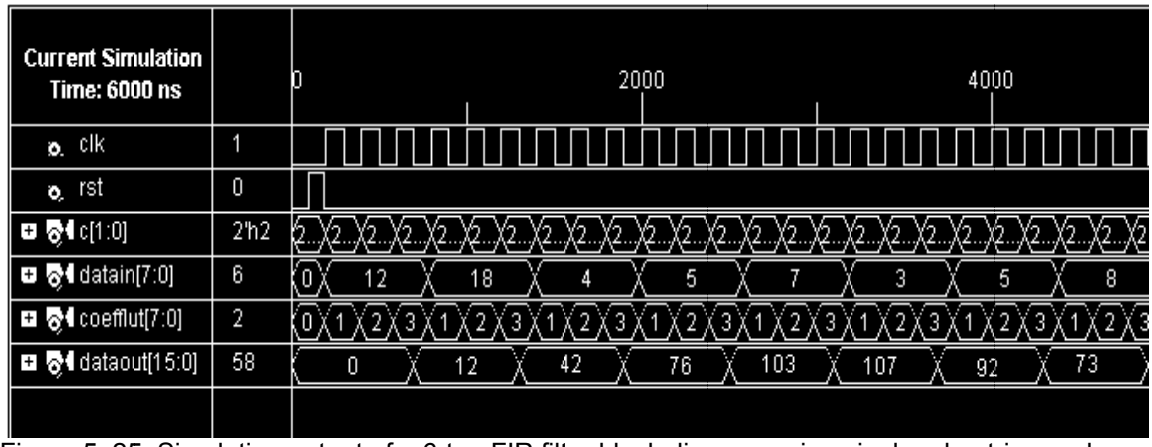


Figure 5. 25: Simulation output of a 6-tap FIR filter block diagram using single edge triggered clocking strategy.

5.6.1.3 Simulation Result of FIR Filter with Dual Edge Triggered Clocking.

Figure 5.26 shows the simulation output of the FIR Filter Block Diagram shown in Fig 5.23. It uses the dual edge triggered level sensitive two phase clocking strategy. It takes more time to produce the output in comparison to the single edge triggered clocking strategy. This is because of multi stage clock gating in a re-arranged fashion which takes a longer path as shown in Fig 5.9. In the simulation output the input port datain takes the input of FIR filter. Another input port coeffLUT accepts the FIR coefficients. The output port dataout gives the output of FIR filter. All the values of these ports are displayed in decimal.

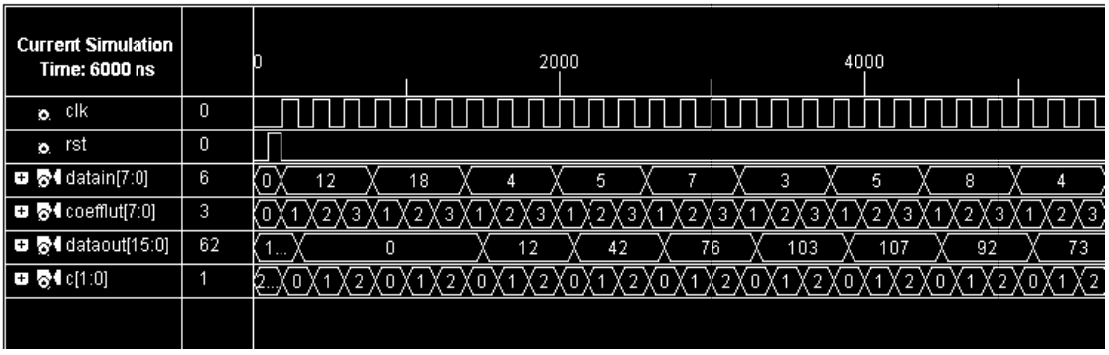


Figure 5. 26: Simulation output of a 6-tap FIR filter block diagram using proposed clocking strategy.

5.6.2 Power Consumption

The power consumed by the two MAC based FIR filter using different clocking strategy are given in table 5.1.

Table 5. 1: Power consumption

FIR Filter structure	Power consumed
Traditional MAC	2.6421 mW
DET based MAC	1.5471 mW

From the above table it is observed that that the DET based MAC FIR filter consumes less power than the traditional one (using SET based MAC). The power saving is approximately 41%. This happens due to the use of dual edge triggering latch that reduces, which consume lot of power. Dual edge triggering facilitates in reducing glitches. The post layout simulation of our proposed latch matches with the circuit level simulation output that is evident from figures 5.19 and 5.21. This validates the proposed design.

5.6.3 Conclusion

The proposed low power latch as shown in figure 5.16 when used in DET based MAC FIR filter, power saving upto 65% is achieved in the filter architecture. The power consumed by different filter architectures discussed in chapter III and IV are compared with the DET based MAC FIR in conjunction with low power latch one, the superiority of this FIR filter architecture is established. Therefore, it is seen that when circuit level low power approaches are added to algorithms, it facilitates for further power reduction. We now proceed to implement this filter architecture in real-time for further validation that is presented in chapter-VI.

Chapter – VI

Real time experimental set up and results for FIR filter using Novel Dual edge triggered Latch

6.1 Introduction

In the previous three chapters we have discuss about 3 types of filters for hearing aid application [P1, P2, and P3]. A comparison is brought out in low power perspective and presented in table 6.1. We can see that FIR filter with dual edge triggered latch consumes less power than others. Therefore, for experimental validation we have chosen MAC based FIR filter with dual edge triggered latch.

Table 6. 1: Power of different filter implemented in Virtex II pro kit.

Comparison between different filters on basis of power consumption in Virtex-II pro board with core voltage 1.2 volts and input speech signal of frequency content 300 Hz to 4 kHz .		
Types of filter		Power consumed
Decimation filter Architecture	5-stage Comb-FIR-FIR	61mW
	5-stage Comb-half band FIR-FIR	50mW
Adaptive hearing aid	Using booth multiplier	40mW
	Using booth Wallace tree multiplier	30mW
FIR Filter structure	Traditional MAC	2.6421mW
	DET based MAC	1.5471mW

The equipments used in this hardware validation are as follows

a. Virtex –II pro kit from Xilinx with AC97 codec.
b. Logic analyzer
c. CRO
d. NI Elvis

For real-time validation we have taken three different signals, these are

a. Voice/ speech signal
b. Voice with music
c. Music only.

The functional block diagram of the experimental setup is shown in figure 6.1 and the complete setup in figure 6.2.

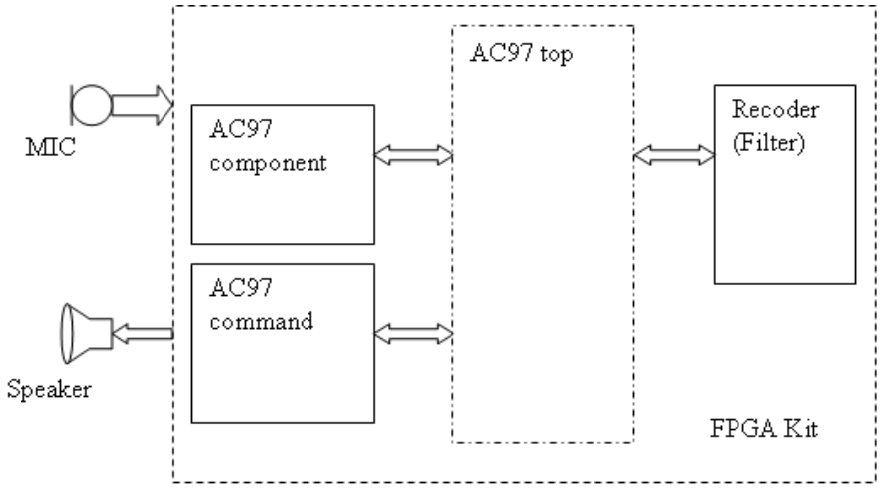


Figure 6. 1: Functional block diagram of the hardware setup

Figure 6. 2: Hardware validation setup for filter implementation in Virtex-II pro board.

6.2 Functional description of the AC'97 codec

The AC'97 codec on Virtex-II Pro board as shown in figure 6.3 is used for providing a path to capture all types of analog signal that can be processed digitally. It features

i. full duplex stereo ADCs and DACs and analog mixers
ii. 4 stereo and 4 mono inputs.
iii. Each mixer input has separate gain, attenuation and mute control
iv. The AC'97 provides a stereo headphone amplifier as one of its stereo outputs.

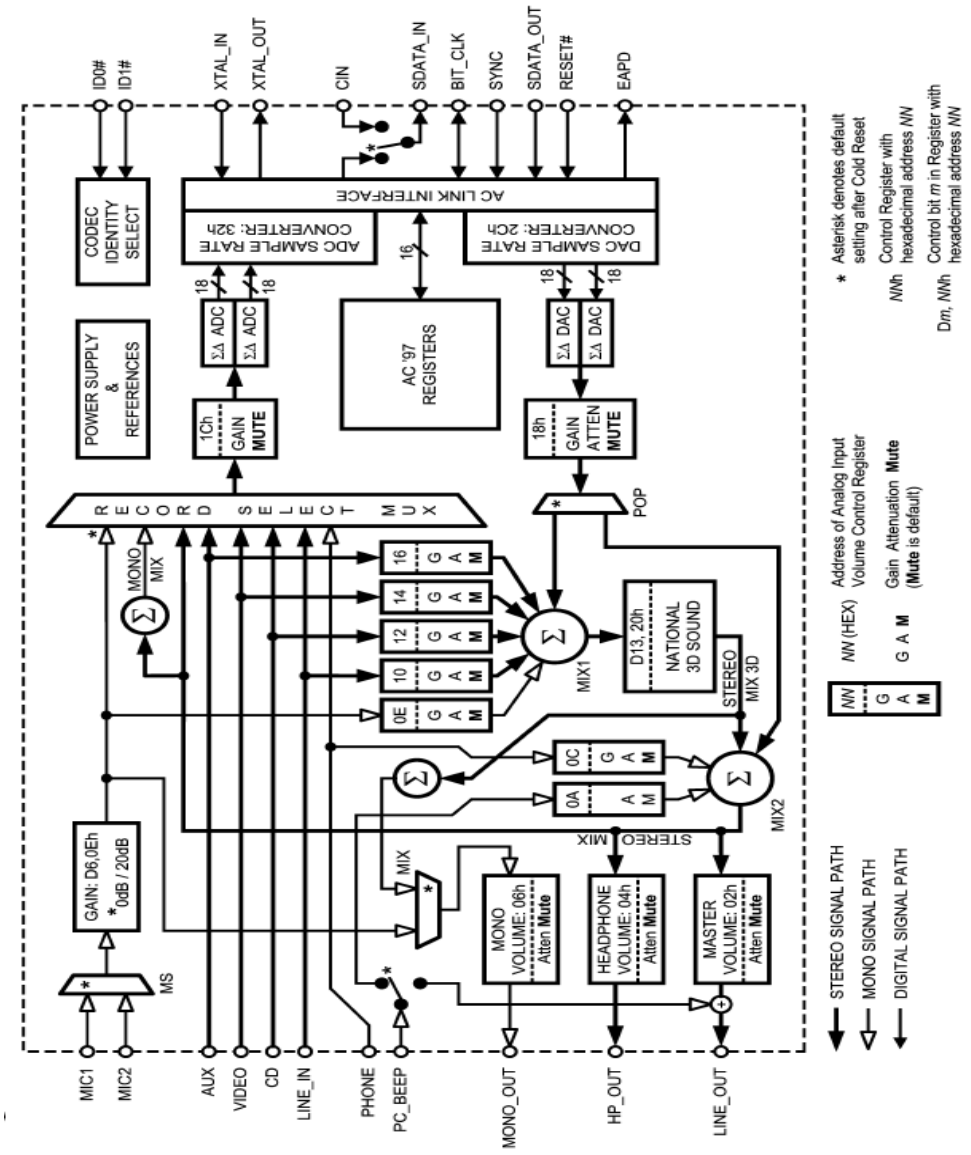


Figure 6. 3: Functional diagram of the AC'97 codec

The sample rate for the ADCs and DACs can be programmed separately with a resolution of 1 Hz to convert any rate in the range 4 kHz – 48 kHz. Sample timing

from the ADCs and sample request timing for the DACs are completely deterministic to ease task scheduling and application software development. These features together with an extended temperature range also make the AC'97 suitable for non-PC codec applications.

The AC '97 architecture separates the analog and digital functions of the PC audio system allowing both for system design flexibility and increased performance.

The AC'97 codec can mix, process and convert among analog (stereo and mono) and digital (AC Link format) inputs and outputs. There are four stereo and four mono analog inputs and two stereo and one mono analog output. A single codec supports data streaming on two input and two output channels of the AC Link digital interface simultaneously. All these are accessed through register addressing methods and the register content for accessing a part is shown in figure 6.4.

All four of the stereo analog inputs and three of the mono analog inputs can be selected for conversion by the 18-bit stereo ADC. Digital output from the left and right channel ADCs is always located in AC Link Input Frame slots 3 and 4 respectively. Input level to either ADC channel can be muted or adjusted from the Record Gain register, 1Ch. Adjustments are in 1.5 dB steps over a gain range of 0 dB to +22.5 dB and both channels mute together.

6.3 Output Frame from AC97

The AC Link Output Frame carries control and PCM data to the AC'97 control registers and stereo DAC. Output Frames are carried on the SDATA_OUT signal which is an output from the AC'97 Digital Controller and an input to the codec. As shown in figure 6.5, Output Frames are constructed from thirteen time slots: one Tag Slot followed by twelve Data Slots. A new Output Frame is signalled with a low-to-high transition of SYNC. SYNC should be clocked from the controller on a rising edge of BIT_CLK and, as shown in figure 6.6 and figure 6.7, the first tag bit in the Frame ("Valid Frame") should be clocked from the controller by the next rising edge of BIT_CLK and sampled by the codec on the following falling edge.

	REG	Name	D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0	Default
	00h	Reset	X	0	0	0	1	1	0	1	0	1	0	1	0	0	0	0	0D50h
Output Volume	02h	Master Volume	Mute	X	X	ML4	ML3	ML2	ML1	ML0	X	X	X	MR4	MR3	MR2	MR1	MR0	8000h
	04h	Headphone Volume	Mute	X	X	ML4	ML3	ML2	ML1	ML0	X	X	X	MR4	MR3	MR2	MR1	MR0	8000h
	06h	Mono Volume	Mute	X	X	X	X	X	X	X	X	X	X	MM4	MM3	MM2	MM1	MM0	8000h
Input Volume	0Ah	PC_Beep Volume	Mute	X	X	X	X	X	X	X	X	X	X	PV3	PV2	PV1	PV0	X	0000h
	0Ch	Phone Volume	Mute	X	X	X	X	X	X	X	X	X	X	GN4	GN3	GN2	GN1	GN0	8008h
	0Eh	Mic Volume	Mute	X	X	X	X	X	X	X	X	20dB	X	GN4	GN3	GN2	GN1	GN0	8008h
	10h	Line In Volume	Mute	X	X	GL4	GL3	GL2	GL1	GL0	X	X	X	GR4	GR3	GR2	GR1	GR0	8808h
	12h	CD Volume	Mute	X	X	GL4	GL3	GL2	GL1	GL0	X	X	X	GR4	GR3	GR2	GR1	GR0	8808h
	14h	Video Volume	Mute	X	X	GL4	GL3	GL2	GL1	GL0	X	X	X	GR4	GR3	GR2	GR1	GR0	8808h
	16h	Aux Volume	Mute	X	X	GL4	GL3	GL2	GL1	GL0	X	X	X	GR4	GR3	GR2	GR1	GR0	8808h
	18h	PCM Out Volume	Mute	X	X	GL4	GL3	GL2	GL1	GL0	X	X	X	GR4	GR3	GR2	GR1	GR0	8808h
ADC Sources	1Ah	Record Select	X	X	X	X	X	SL2	SL1	SL0	X	X	X	X	X	SR2	SR1	SR0	0000h
	1Ch	Record Gain	Mute	X	X	X	GL3	GL2	GL1	GL0	X	X	X	X	GR3	GR2	GR1	GR0	8000h
	20h	General Purpose	POP	X	3D	X	X	X	MIX	MS	LPBK	X	X	X	X	X	X	X	0000h
	22h	3D Control (Read Only)	X	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0101h
X	24h	Reserved	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0000h
	26h	Powerdown Ctrl/Stat	EAPD	PR6	PR5	PR4	PR3	PR2	PR1	PR0	X	X	X	X	X	X	DAC	ADC	000Xh
	28h	Extended Audio ID	ID1	ID0	X	X	X	X	AMAP	0	0	0	X	X	0	X	0	VRA	X201h
	2Ah	Extended Audio Control/Status	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	VRA	0000h
	2Ch	PCM DAC Rate	SR15	SR14	SR13	SR12	SR11	SR10	SR9	SR8	SR7	SR6	SR5	SR4	SR3	SR2	SR1	SR0	BB80h
	32h	PCM ADC Rate	SR15	SR14	SR13	SR12	SR11	SR10	SR9	SR8	SR7	SR6	SR5	SR4	SR3	SR2	SR1	SR0	BB80h
X	5Ah	Vendor Reserved 1	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0000h
	74h	Chain-In Control	X	X	X	X	X	X	X	X	X	X	X	X	X	X	ID1	ID0	000Xh
X	7Ah	Vendor Reserved 2	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	0000h
	7Ch	Vendor ID1	0	1	0	0	1	1	1	1	0	1	0	1	0	0	1	1	4E53h
	7Eh	Vendor ID2	0	1	0	0	0	0	1	1	0	1	0	1	0	0	0	0	4350h

Figure 6. 4: Register content for accessing different sections of a ac97 codec

AC Link Serial Interface Protocol

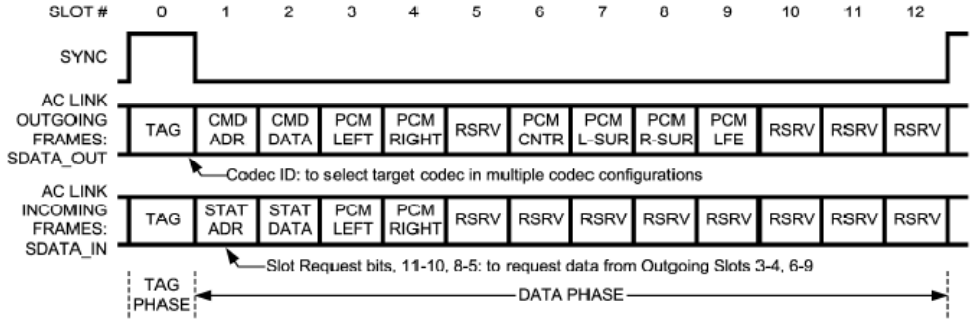


Figure 6. 5: AC link output frames for ac97

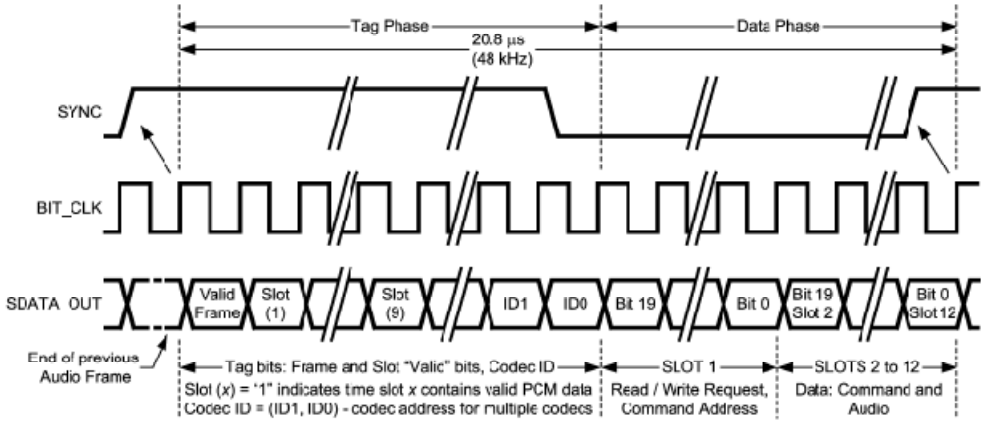


Figure 6. 6: Structure of output frame

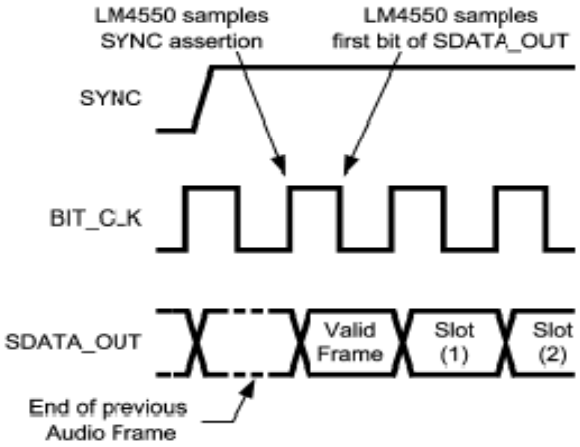


Figure 6. 7: Start of a output frame

Each Frame consists of 256 bits with each of the twelve Data Slots containing 20 bits. Input and Output Frames are aligned to the same SYNC transition. The Codec checks each Frame to ensure 256 bits are received. If a new Frame is detected (a low-to-high transition on SYNC) before 256 bits are received from the old Frame then the new Frame is ignored *i.e.* the data on SDATA_OUT is discarded until a valid new frame is detected. The LM4550 expects to receive data MSB first, in an MSB justified format.

The first bit of Slot 0 is designated the "Valid Frame" bit. If this bit is 1, it indicates that the current Output Frame contains at least one slot of valid data and the codec will check further tag bits for valid data in the expected Data Slots. Since it is a two channel codec the codec can only receive data from four slots in a given frame and so only checks the valid-data bits for 4 slots. In Primary mode these tag bits are for: slot 1 (Command Address), slot 2 (Command Data), slot 3 (PCM data for left DAC) and slot 4 (PCM data for right DAC). The last two bits in the Tag contain the

Codec ID used to select the target codec to receive the frame in multiple codec systems. When the frame is being sent to a codec in one of the Secondary modes the controller does not use bits 14 and 13 to indicate valid Command Address and Data in slots 1 and 2.

Slot 1 is used by a controller to indicate both the address of a target register in the codec and whether the access operation is a register read or register write. The MSB of slot 1 (bit 19) is set to '1' that indicates current access operation is 'read'. Bits 18 through 12 are used to specify the 7-bit register address of the read or write operation and all other bits are set to '0'. Slot 2 is used to transmit 16-bit control data to the codec when the access operation is 'write'. The least significant four bits should be stuffed with zeros by the AC '97 controller. If the access operation is a register read, the entire slot, bits 19 through 0 should be stuffed with zeros. Slots 3 and 4 are 20-bit fields used to transmit PCM data to the left and right channels of the stereo DAC when the codec is in Primary mode or Secondary mode 1. Any unused bits should be stuffed with zeros. Slots 7 and 8 are 20-bit fields used to transmit PCM data to the left and right channels of the stereo DAC when the codec is in Secondary mode 2. Any unused bits should be stuffed with zeros.

Slots 6 and 9 are 20-bit fields used to transmit PCM data to the left and right channels of the stereo DAC when the codec is in Secondary mode 3. Any unused bits should be stuffed with zeros. Slots 5, 10, 11, 12 – reserved slots, are not used by the codec and should all be stuffed with zeros by the AC '97 Controller.

6.4 AC Link Input Frame

The AC Link Input Frame contains status and PCM data from the codec control registers and stereo ADC. Input Frames are carried on the SDATA_IN signal which is an input to the AC '97 Digital Audio Controller and an output from the codec. As shown in figure 6.8, Input Frames are constructed from thirteen time slots: one Tag Slot followed by twelve Data Slots. The Tag Slot, Slot 0, contains 16 bits of which 5 are used by the codec. One is used to indicate that the AC Link interface is fully operational and the other 4 to indicate the validity of the data in the four of the twelve following Data Slots that are used by the codec. Each Frame consists of 256 bits with each of the twelve data slots containing 20 bits as shown in figure 8.

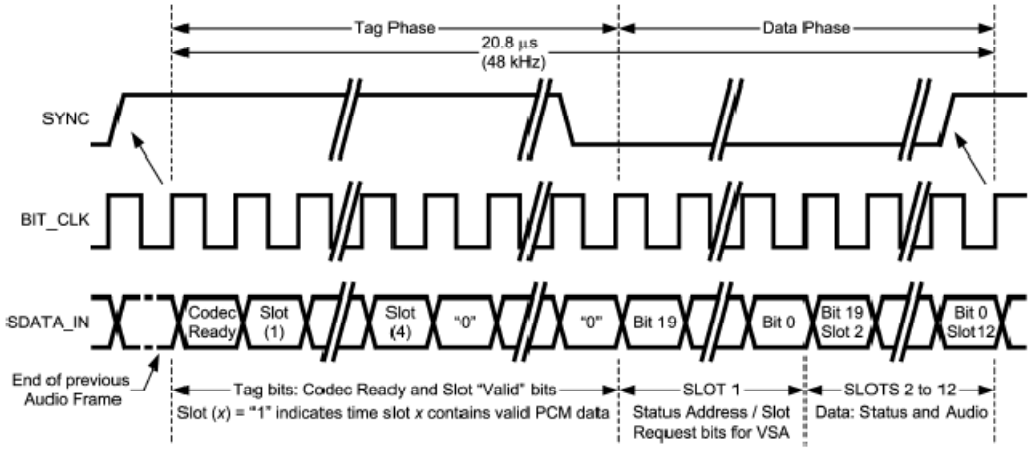


Figure 6. 8: AC link input frame

A new Input Frame is signalled with a low-to-high transition of SYNC. SYNC should be clocked from the controller on a rising edge of BIT_CLK and, as shown in figure 6.6 and figure 6.7, the first tag bit in the Frame ("Codec Ready") is clocked from the codec by the next rising edge of BIT- _CLK. The codec always clocks data to SDATA_IN on a rising edge of BIT_CLK and the controller is expected to sample SDATA_IN on the next falling edge. The codec samples SYNC on the rising edge of BIT_CLK. Input and Output Frames are aligned to the same SYNC transition.

The codec checks each Frame to ensure 256 bits are received. If a new Frame is detected (a low-to-high transition on SYNC) before 256 bits are received from an old Frame then the new Frame is ignored *i.e.* no valid data is sent on SDATA_IN until a valid new Frame is detected as sown in figure 6.9. The codec transmits data MSB first, in an MSB justified format. All reserved bits and slots are stuffed with "0"s by the codec.

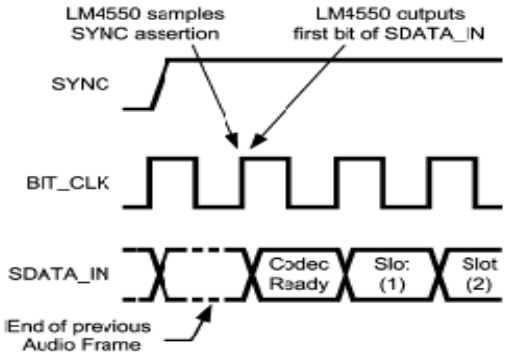


Figure 6. 9: Start of input frame

The first bit (bit 15, "Codec Ready") of slot 0 in the AC Link Input Frame indicates when the codec's AC Link digital interface and its status/control registers are fully operational. The digital controller is then able to read the LSBs from the Power

down Control/Stat register (26h) to determine the status of the four main analog sub-sections. Slot 1 echoes (in bits 18 – 12) the 7-bit address of the codec control/status register received from the controller as part of a read-request in the previous frame. If no read request was received, the codec stuffs these bits with zeros. The 6 bits 11, 10, 8 – 5 are Slot Request bits that support the Variable Rate Audio (VRA) capabilities of the codec. The codec uses bits 11 and 10 to request DAC data from these two slots. If bits 11 and 10 are set to 0, the controller should respond with valid PCM data in slots 3 and 4 of the next Output Frame.

If bits 11 and 10 are set to 1, the controller should not send data. Similarly, if the codec is in Secondary mode 2, bits 7 and 6 are used to request data from slots 7 and 8 in the Output Frame. If in Secondary mode 3, bits 8 and 5 request data from slots 6 and 9. The codec has full control of the slot request bits. By default, data is requested in every frame, corresponding to a sample rate equal to the frame rate (SYNC frequency) – 48 kHz when XTAL_IN = 24.576 MHz.

This slot 2 returns 16-bit status data read from a codec control/ status register. The codec sends the data in the frame following a read-request by the controller (bit 15, slot 1 of the Output Frame). If no read-request was made in the previous frame the codec will stuff this slot with zeros.

This slot3 contains sampled data from the left channel of the stereo ADC. The signal to be digitized is selected using the Record Select register (1Ah) and subsequently routed through the Record Select Mux and the Record Gain amplifier to the ADC. This is a 20-bit slot and the digitized 18-bit PCM data is transmitted in an MSB justified format. The remaining 2 LSBs are stuffed with zeros.

This slot 4 contains sampled data from the right channel of the stereo ADC. The signal to be digitized is selected using the Record Select register (1Ah) and subsequently routed through the Record Select Mux and the Record Gain amplifier to the ADC. This is a 20-bit slot and the digitized 18-bit PCM data is transmitted in an MSB justified format. The remaining 2 LSBs are stuffed with zeros. Slots 5 – 12 of the AC Link Input Frame are not used for data by the codec and are always stuffed with zeros.

6.5 Structure of the filter implemented

We have taken the fir filter for hardware implementation as shown in figure 6.10. This figure shows a 6 tap FDF FIR filter. In this the counter is used to select the taps and also the corresponding coefficient from the memory ‘Coeff LUT’. The data selected

from the tap registers are pre-added and then converted into sign magnitude number. After conversion it is fed to the slave triggered latch (shown crossed) which is then multiplied by the coefficient. Before multiplication, the coefficient is also passed through the slave triggered latch. After multiplication it is again converted back into 2's complement number. The data is then passed into an accumulator whose output feed back to input through two latches as shown in figure. Accumulator output is also fed to output through two latches.

The accumulator output is given as

$$Dout(n)=Dout(n-1)+multout \quad (6.1)$$

where Dout= output of the accumulator

multout=multiplication output.

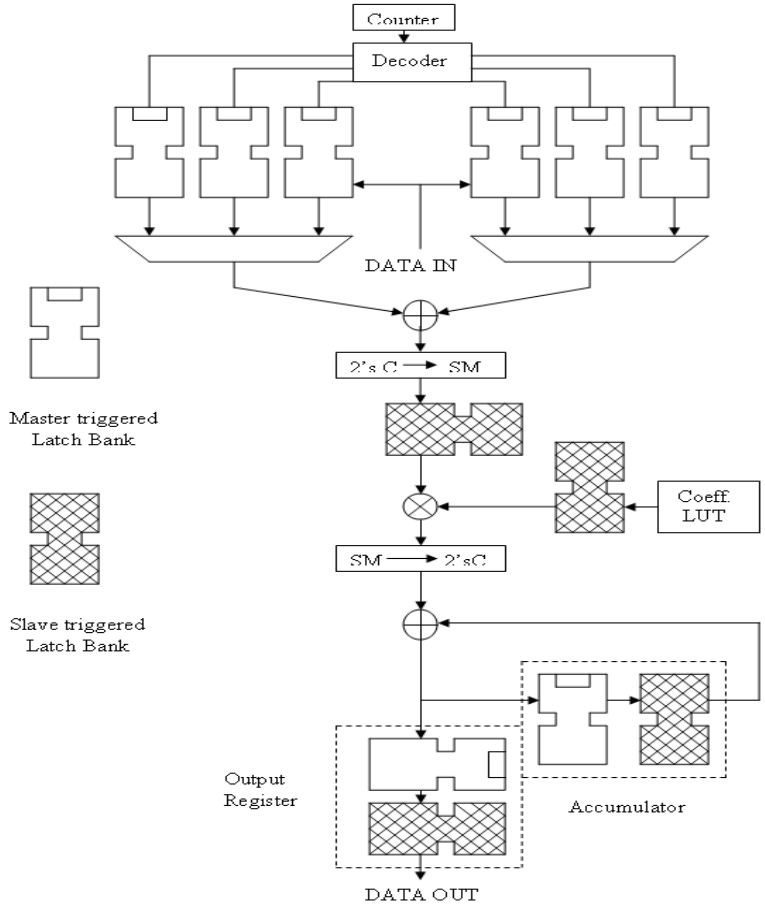


Figure 6. 10: FIR filter structure with dual edge triggering for hardware implantation

6.6 Results and conclusion

For real time validation of the filter, we have taken three different types of signals mainly

a. Speech signal
b. Voice with Music
c. Music only.

For each signal three sets of reading are taken and there amplitude spectrums are shown below in following figures.

1. The logic analyser plot shows the input signal and output response (in form of data) of the filter. The clock signal is of 100 MHz and bit clock signal is of 27 MHz is also shown but are dark due to high frequency (figure 6.11).

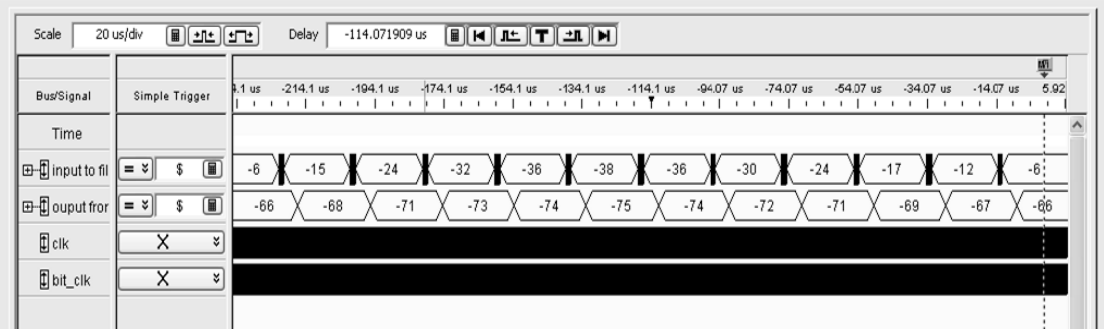


Figure 6. 11: Spectrum analyser output of input and output data

2. Amplitude spectrum of the various input and output signals are shown figure below captured using National Instrument's ELVIS and LabView.

a) Speech signal

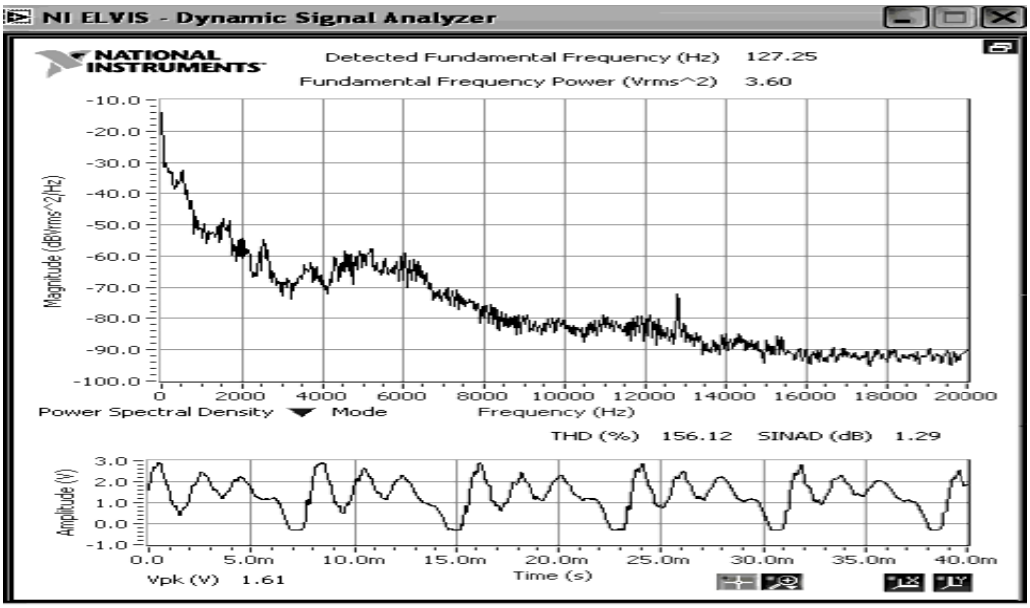


Figure 6. 12: Spectrum of speech signal without filter implementation

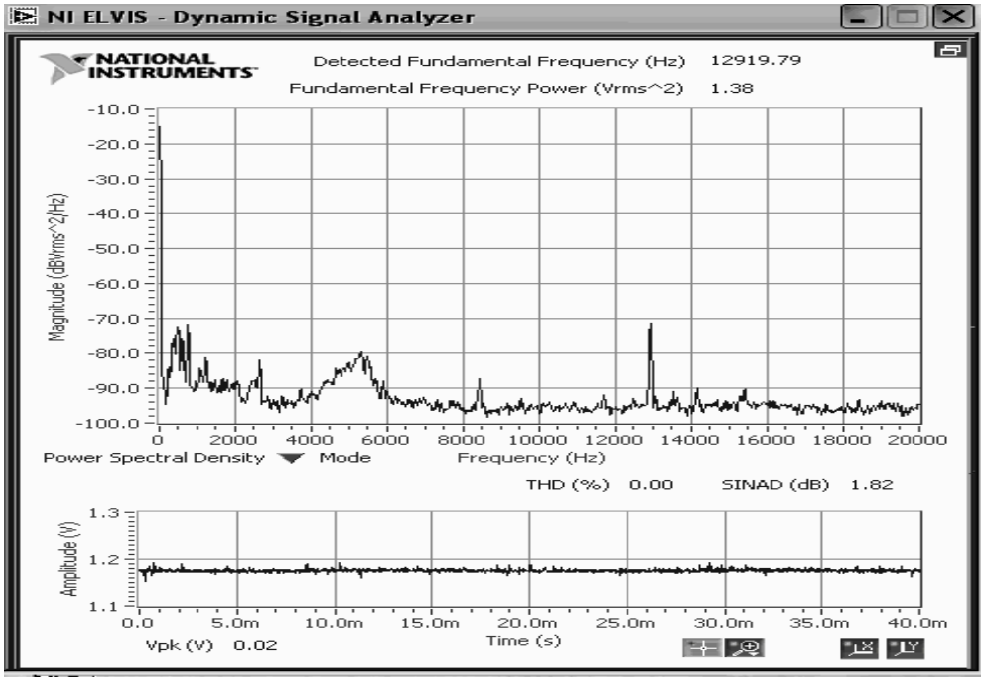


Figure 6. 13: Amplitude spectrum of the speech signal after filtering.

The filter was tested with a speech signal of self whose amplitude spectrum are shown in figure 6.12 without filtering and figure 6.13 shows the output of the filter when speech is passed. From the figure we can see that some of the noise is suppressed.

b) Music only

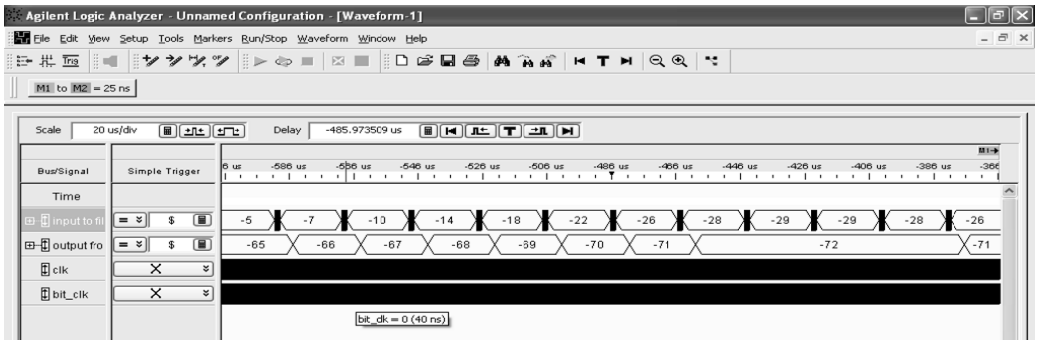


Figure 6. 14: logic analyzer output of the filter for music signal only.

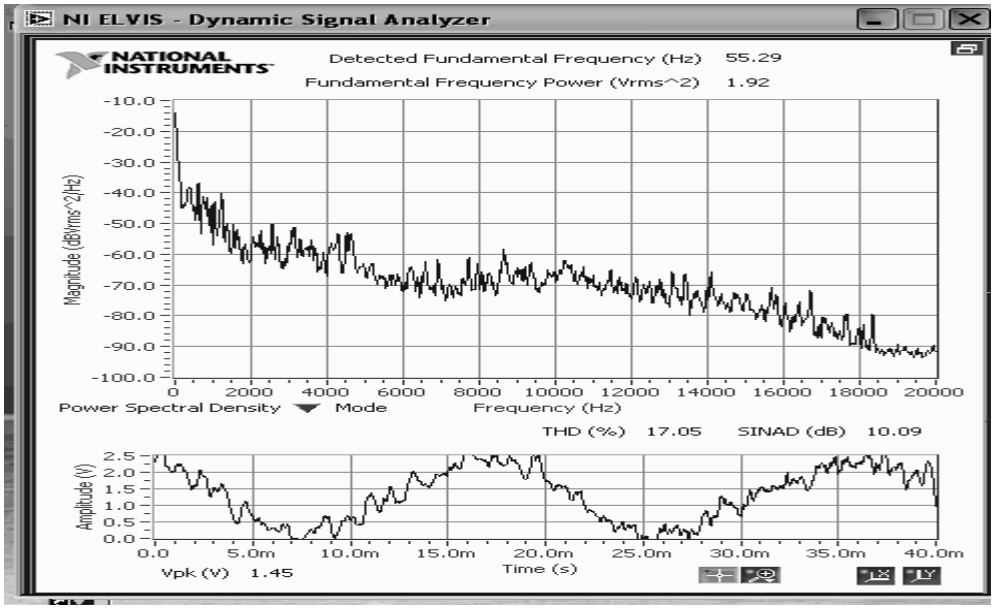


Figure 6. 15: Amplitude spectrum of the music signal taken

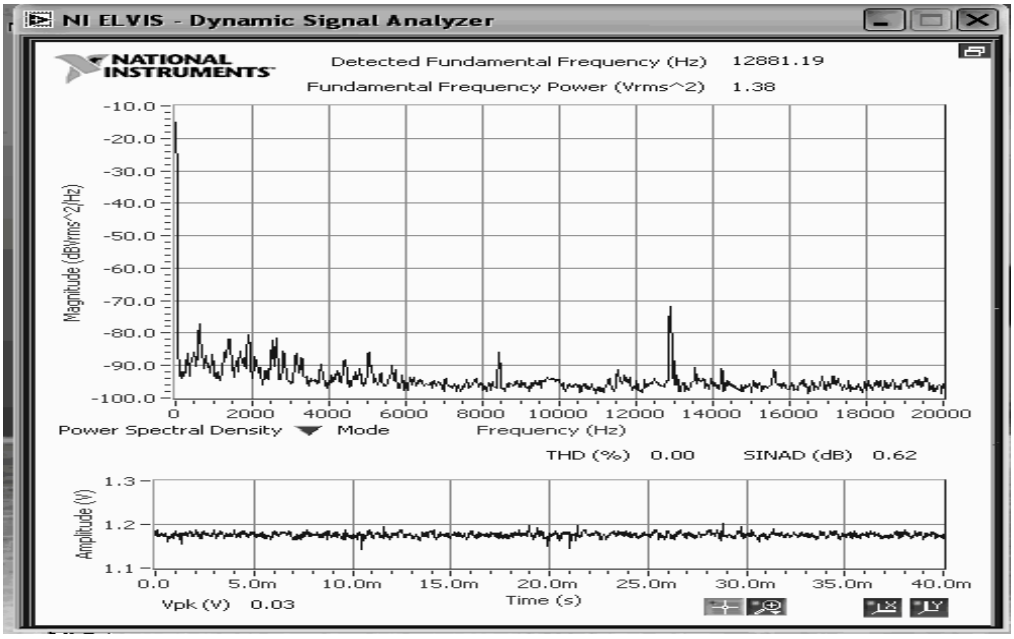


Figure 6. 16: Amplitude spectrum of the music signal after filtering.

The filter was tested with a music signal whose amplitude spectrum are shown in figure 6.15 without filtering and figure 6.16 shows the output of the filter when speech is passed From the figure we can see that some of the noise is suppressed.

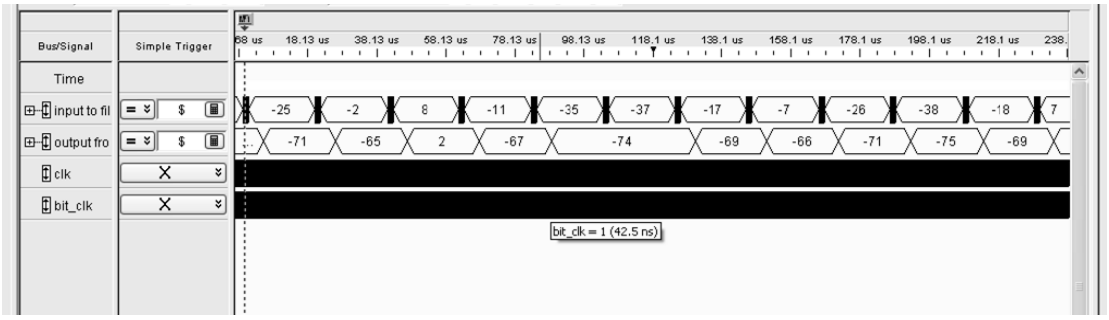


Figure 6. 17: Output of the logic analyser after filtering the signal(music and voice).

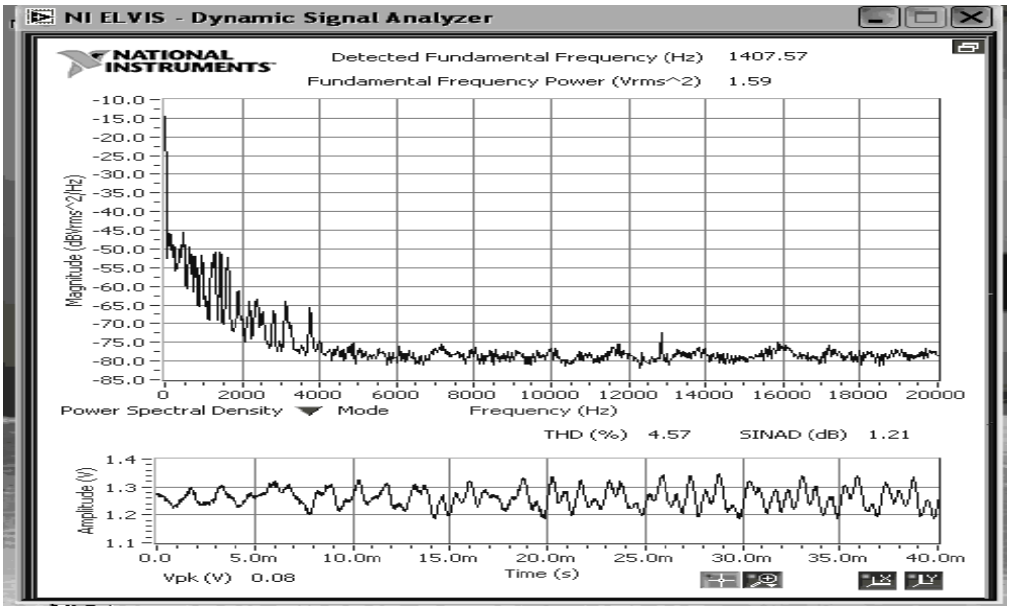


Figure 6. 18: Amplitude spectrum of a signal containing music and voice.

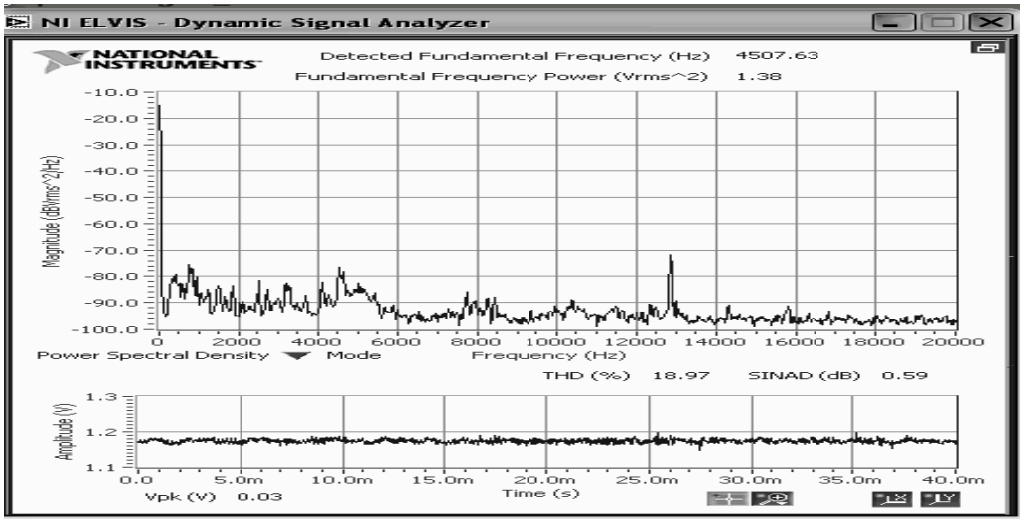


Figure 6. 19: Amplitude spectrum of the signal after filtering the signal (music and voice).

The filter was tested with a speech signal whose amplitude spectrum are shown in figure 6.18 without filtering and figure 6.19 shows the output of the filter

when voice plus music is passed. From the figure we can see that some of the noise is suppressed at the out put of the filter.

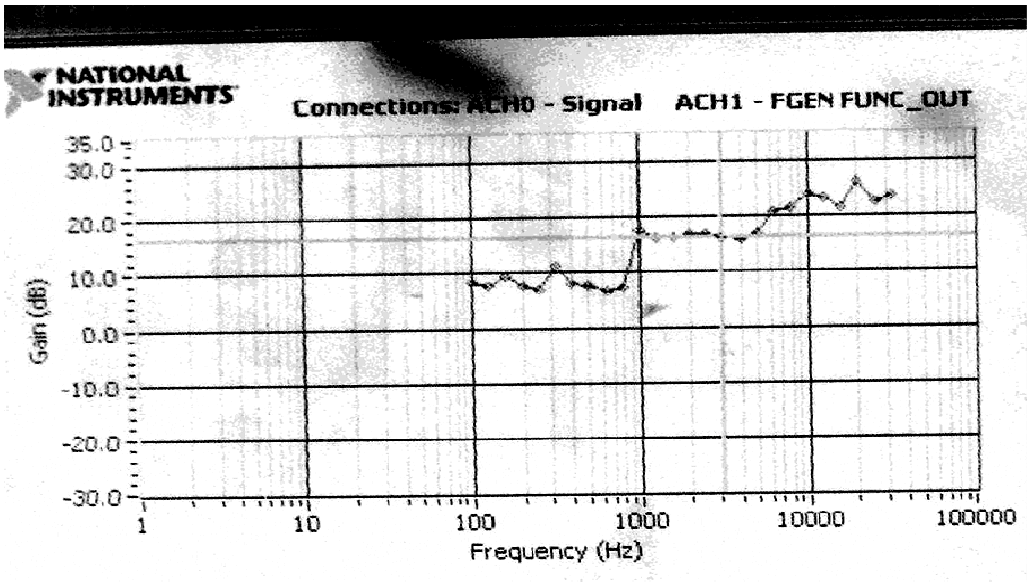


Figure 6. 20: Gain plot of the amplifier

6.7 Conclusion

From above figures (from figure 6.1 to figure 6.19) it can be seen that filter is working perfectly for all type of signals and from figure 6.20 it can be seen that a maximum gain that can be obtained is about 27dB which is suitable for hearing aid application. Therefore a hearing aid device using FIR filter and dual edge triggered latch is found to best as far as low power is concerned, besides providing necessary filtering characteristics and amplification.

Chapter - VII

Conclusion

7.1 Introduction

The general conclusion drawn from this thesis and some suggested new directions are presented in this paper. The objective of this thesis is to design suitable low power digital filters for hearing aid applications. The algorithms and the filter structures are to be chosen that are hardware realizable; low power being the most important constraint; suitable low power VLSI techniques must be applied in each case to minimize power. All the digital filters should be implemented in FPGA to test functionality.

7.2 Conclusions

The work done in this thesis to achieve above mentioned objectives is summarized below.

1. Decimation filter: A suitable filter for hearing aid application is designed using 'Distributed Arithmetic'. The filter uses decimation filter followed by two filters i.e. FIR filter followed by corrector filter. Using Comb-FIR-FIR and Comb-Half-band FIR-FIR, the power is calculated and the values are 61 mW and 50 mW respectively. From these values it is observed that the Comb-half-band FIR-FIR structure consumes less power than the former but still it is high due to the fact that filters are highly serial and latency is also vey high. However use of 'Distributed Arithmetic' in decimation filter certainly facilitates reduction of power compared to CSD (Canonical Signed Digit) approach already adopted in the literature.
2. Adaptive Filter: We have implemented a digital hearing aid using Booth-Wallace tree multiplier and adaptive algorithm which is basically a lattice filter. This filter after implemented in hardware successfully is tested for its power consumption. The power consumption is calculated using Xilinx Xpower which comes about 30 mW. It shows less power consumption than the decimation filter but still power consumption is not quite low. An adaptive filter using Booth-Wallace tree multiplier shows a reduction in power consumption when compared to a similar filter that uses Booth multiplier that consumes 40 mW power on the same platform.

3. Algorithmic and Circuit level integration for low power VLSI design: We have used algorithmic method for low power filter implementation for Decimation filter and Adaptive filter case. However power consumption is not low considering the fact that the filters are to be used in a hearing aid application. A combination of both algorithmic level and circuit level for low power VLSI design reduces the power consumption in a digital filter. In VLSI circuit design, the clock consumes a lot of power due to the existence of clock jitters and clock skew. Clock gating principle is applied to achieve low power. A novel low power latch using 10 transistors is designed. We have taken MAC based FDF FIR structure that reduces number of multiplications which facilitates power reduction. This filter is implemented using the designed low power latch and the clock gating. The power consumed by the MAC based FDF FIR filter is about 2.6421 mW and 1.5471 mW for SET and DET latches respectively. It is observed that a significant power reduction is achieved using DET based MAC based FDF FIR filter. Therefore integration of circuit level approach and the algorithmic approach facilitates further reduction in power dissipation in the digital filter.
4. Low power latch: A novel low power latch is designed in this dissertation. The design is carried out using both 350 nm and 180 nm technology. The functionality is tested and found to be correct. Layout and post-layout simulation confirms the functionality and the power consumption in the latch. This latch is used as circuit component in the FDF FIR filter that facilitates reduction of power consumption in the filter
5. Clock gating: There exists a lot of scope in reduction of power in algorithmic and circuit level implementation. A proper application of the clock gating principle reduces power in VLSI circuit. This principle is applied to the designed low power latch and significant power reduction is achieved. The power consumed by the latch is about 22 μW when it is driven at 250 MHz clock.

6. Real-time hardware Validation: All the filter structures are implemented in FPGA and power dissipation is calculated for all cases. We observed that the FDF FIR filter with dual edge triggering latch and clocking strategy consumes lowest power while compared among the filters that are considered. We have chosen this filter for real time validation and this is tested for real-time experimentation. The filter provides necessary amplification and frequency response in case of speech, music, and voice with music signal. This filter provides a gain of 27 dB (maximum) that matches with most of the hearing aid (manufacturer's) specifications. Hence it can be concluded that FDF FIR digital filter in conjunction with low power latch is strong candidate for hearing application.

7.3 Scope for further Work

Further study can be carried in following areas

1. Detailed power calculations at different abstraction level can be done using Prime power from Synopsys and hence possibility of power reduction at different abstraction level can be considered. Further more, a trade-off between area, speed and power can be brought satisfying power constraint.
2. ASIC design for the low power digital filter (FDF-FIR) with low power latch and clock gating can be carried out.
3. A more robust filter (suitable for hearing aid application) can be investigated and VLSI design can be carried out.

References

[1] http://www.nidcd.nih.gov/health/hearing/hearingaid.asp

[2] http://www.hearingresearch.org/Dr.Ross/Feedback-Cancellation.htm

[3] Sanjit K. Mitra, "DSP A Computer based Approach", Tata McGraw Hill Publication, 2nd Edition, 2001.

[4] Ravi K. Kolagotla, Jose Fridman, Bradley C. Aldrich, Marc M. Hoffman, William C. Anderson, Michael S. Allen, David B. Witt, Randy R. Dunton, And Lawrence A. Booth, Jr., "High Performance Dual-Mac DSP Architecture", IEEE Signal Processing Magazine, July, 2002, pp. 42-53.

[5] James M. Kates, "Feedback Cancellation in Hearing Aids Using Constrained Adaptation" , Workshop on Applications of Signal Processing to Audio and Acoustics, New Pulfz, New York, Oct. 17-20, 1999.

[6] Neeraj Magotra Juan G. Vargas-Rubio, "Comparison Of Two Open-Loop Adaptive Speech Enhancement Algorithms for Digit Al Hearing Aids", IS-CAS 2000 - IEEE International Symposium on Circuits and Systems, May 28-31, 2000, Geneva, Switzerland.

[7] Marcio G. Siqueira, Abeer Alwan, "Steady-State Analysis of Continuous Adaptation in Acoustic Feedback Reduction Systems for Hearing-Aids" IEEE Transactions on Speech and Audio Processing, Vol. 8, No. 4, July 2000 pp. 443-453.

[8] Boaz Rafaely and Mariano Roccasalva-Firenze, "Control of Feedback in Hearing Aids—A Robust Filter Design Approach", IEEE Transactions On Speech And Audio Processing, Vol. 8, No. 6, pp. 754-756, November 2000.

[9] Johan Hellgren and Urban Forssell, " Bias of Feedback Cancellation Algorithms in Hearing Aids Based on Direct Closed Loop Identification", IEEE Transactions on Speech and Audio Processing, Vol. 9, No. 7, pp. 906-913, November 2001.

[10] Stefan Gustafsson, Rainer Martin, Peter Jax, *a*nd Peter Vary, " A Psychoacoustic Approach To Combined Acoustic Echo Cancellation And Noise Reduction", Ieee Transactions On Speech And Audio Processing, Vol. 10, No. 5, July 2002, pp. 245- 256.

[11] Yong Lia, and Ying Wei "A Computationally Efficient Nonuniform FIR Digital Filter Bank For Hearing Aids", IEEE Transaction on Circuits and Systems, Vol. 52, No. 12, December, 2005, pp. 2754 – 2762.

[12] Thomas Fillon and Jacques Prado, "Acoustic Feedback Cancellation For Hearing-Aids, Using Multi-Delay Filter", 5th Nordic Signal Processing Symposium, October 4-7, 2002, on board Hurtigruten, Norway.

[13] Johan Hellgren "Analysis of Feedback Cancellation in Hearing Aids With Filtered-X LMS and the Direct Method of Closed Loop Identification", IEEE Transactions on Speech and Audio Processing, Vol. 10, No. 2, pp. 119-131, February, 2002.

[14] Tomas Gänsler and Jacob Benesty, "New Insights Into the Stereophonic Acoustic Echo Cancellation Problem and an Adaptive Nonlinearity Solution", IEEE Transactions On Speech And Audio Processing, Vol. 10, No. 5, July 2002 257, pp. 257-267.

[15] Young-cheol Park, In-young Kim and Sang-min Lee "An efficient adaptive feedback cancellation for hearing aids", Proceedings of the 25th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society, 2003, Vol. 2, pp. 1647- 1650.

[16] Hsiang-Feng Chi, Shawn X. Gao, Sigfrid D. Soli and Abeer Alwan "Band-limited feedback cancellation with a modified filtered-X LMS algorithm for hearing aids", Journal of Speech Communication, Elsevier, Vol. 39, Issue -2, January 2003, pp. 147-161.

[17] Ann Spriet, Marc Moonen and Jan Wouters "Stochastic Gradient-Based Implementation of Spatially Preprocessed Speech Distortion Weighted Multichannel Wiener Filtering for Noise Reduction in Hearing Aids " IEEE Transactions on Signal Processing, Vol. 53, No. 3, March 2005 911, pp. 911-925.

[18] Ngwa Abinwi Shusina and Boaz Rafaely "Unbiased Adaptive Feedback Cancellation in Hearing Aids by Closed-Loop Identification", IEEE Transactions On Audio, Speech, And Language Processing, Vol. 14, No. 2, March 2006, pp. 658-665.

[19] Andy W. H. Khong and Patrick A. Naylor, "Stereophonic Acoustic Echo Cancellation Employing Selective-Tap Adaptive Algorithms", IEEE Transactions

on Audio, Speech, And Language Processing, Vol. 14, No. 3, May 2006 pp. 785- 796.

[20] A Ben Hamida, "An adjustable filter-bank based algorithm for hearing aid systems" IECON-99, Vol. 3, pp. 1187-1192.

[21] F. Carbognani, F. Bürgin, L. Henzen, H. Koch, H. Magdassian, C. Pedretti, H. Kaeslin, N. Felber, W. Fichtner, "A 0.67-mm2 45-µW DSP VLSI Implementation of an Adaptive Directional Microphone for Hearing Aids", European Conference on Circuit Theory and Design (ECCTD), Cork, Ireland, 29. Aug - 1. Sep 2005.

[22] S. Sarkar, A. Basu, A.K. Majumdar, "Representation and synthesis of interface of a circuit for its reuse [VLSI design]", Proceedings of Ninth International Conference on VLSI Design, 1996, 3-6 Jan. 1996, pp.140 - 145

[23] L.C.C. Marques, C. Galup-Montoro, S.N. Filho, M.C. Schneider, "A Switched-Mosfet Filter for Application in Hearing Aid Devices", IEEE International Symposium on Circuits and Systems, May**, 2001,** Vol. 1, pp. 77-80.(old 21)

[24] C.-H. Kao, W.-P. Lin and W.-C. Chen, "High efficiency and low distortion switching power amplifier for hearing aids", IEE Proceedings on Circuits, Devices and Systems, April 2006, vol. 153, issue – 2, pp. 143-147.

[25] Ultra-Low-Power DLMS Adaptive Filter for Hearing Aid Applications", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Dec, 2003, vol. 11, issue- 6, pp. 1058-1067.

[26] Daniel J. Allred, Walter Huang, Venkatesh Krishnan, Heejong Yoo, and David V. Anderson, "An FPGA Implementation for a High Throughput Adaptive Filter Using Distributed Arithmetic", Proceedings of the 12th Annual IEEE Symposium on Field-Programmable Custom Computing Machines (FCCM'04), 2004.

[27] A.T. Erdogan, M. Hasan and T. Arslan, "Algorithmic low power FIR cores", IEE Proceedings on Circuits, Devices and Systems, June, 2003, vol. 150, issue-3, pp. 155-160.

[28] R. H. Turner and R. F. Woods, "Highly Efficient, Limited Range Multipliers for LUT-Based FPGA Architectures", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Oct. 2004, vol. 12, issue – 10, pp. 1113-1118.

[29] K.A. Vinger and J. Torresen, "Implementing evolution of FIR-filters efficiently in an FPGA", Proceedings of NASA/DoD Conference on Evolvable Hardware, July, 2003, pp. 26-29.

[30] A.T. Erdogan and T. Arslan, "Low power block based FIR filtering cores", Proceedings of the 2003 International Symposium on Circuits and Systems (ISCAS '03), vol. 5, pp. V-341 to V-344.

[31] Kim Kyung-Saeng and Lee Kwyro, "Low-power and area-efficient FIR filter implementation suitable for multiple taps" IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Feb 2003, vol. 11, issue 1, pp. 150-153.

[32] Behrooz Parhami and Ding-Ming Kwai, **"**Parallel Architectures and Adaptation Algorithms for Programmable FIR Digital Filters with Fully Pipelined Data and Control Flows", Journal of Information Science and Engineering, January 2003, Vol.19 No.1, pp.59-74.

[33] A.T. Erdogan, E. Zwyssig and T. Arslan**,** "Architectural trade-offs in the design of low power FIR filtering cores", IEE Proc.-Circuits Devices Syst., Vol. 151, No. 1, February 2004, pp. 10-17.

[34] Vivek Venugopal, Khalid H. Abed Shailesh B. Nerurkar, "Design and Implementation of a Decimation filter for Hearing Aid Applications", Proceedings of IEEE Southeast Con., April, 2005, pp. 111-115.

[35] Mitsuru Yamada and Akinori Nishihara, "High-speed FIR Digital Filter with CSD Coefficients Implemented on FPGA", Proceedings of the ASP-DAC (Design Automation Conference) Asia and South Pacific, 2001, pp. 7-8.

[36] E. P. Zwyssig, A.T. Erdogan, T. Arslan "Low Power System On Chip Implementation Scheme Of Digital Filtering Cores" IEE Seminar on Low Power IC Design, 2001, pp. 5/1-5/9.

[37] Masahide Abe, Hiroki Arai and Masayuki Kawamata, "Design and FPGA Implementation of a Structure of Evolutionary Digital Filters for Hardware Implementation", IEEE International Symposium on Circuits and Systems, 2005(ISCAS 2005), pp. 528-531.

[38] A.T. Erdogan and T. Arslan, "A Coefficient Segmentation Algorithm For Low Power Implementation Of Fir Filters", IET Electronics Letter, Sept. 1998, Vol. 34, issue- 19, pp. 1817-1819.

[39] Kavita Nair and Ramesh Harjani, "Compact, Ultra Low Power, Programmable Continuous-Time Filter Banks for Feedback Cancellation in Hearing Aid", Proceedings of 12th International Conference on VLSI Design, 1999, pp. 55-60.

[40] S. Narayanaswamy, S. Seshan, E. Amir, et al., "A low-power, lightweight unit to provide ubiquitous information access application and network support for InfoPad", IEEE Personal Communications, pp. 4–17, Apr. 1996.ch2-3

[41] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design", IEEE Journal of Solid-State Circuits, vol. 27, no. 4, pp. 473–483, Apr. 1992.ch2-4

[42] A. Chadrakasan and R. W. Brodersen, Low Power Digital CMOS Design, Kluwer, 1995.ch2-5

[43] A. Chadrakasan, M. P. Potkonjak, R. Mehra, J. Rabey, and R. W. Brodersen, "Optimizing power using transformations," IEEE Trans. on Computer-Aided Design, Vol. 14, No. 1, pp. 12–31, Jan., 1995.ch2-6

[44] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design", IEEE Journal of Solid-State Circuits, vol. 27, no. 4, pp. 473–483, Apr. 1992.ch3-7

[45] G. K Yeap, "Practical Low Power Digital VLSI Design", Kluwer Academic Publishers, Norwell, Mass., 1998.ch2-8

[46] H. J. M. Veendrick, "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits," IEEE Journal of Solid-State Circuit, Vol. 19, pp. 468–473, Aug., 1984.ch2-9

[47] L. Nielsen, C. Nielsen, J. Spars, and K. van Berkel, "Lowpower operation using self-timed circuits and adaptive scaling of the supply voltage," IEEE Trans. on VLSI Systems, Vol. 2, No. 4, pp. 391–397, Dec., 1994.ch2-10

[48] J. M. Rabaey and M. Pedram, "Low Power Design Methodologies", Kluwer Publishers, 1996, ch2-11

[49] Sakurai, T.; "Low-power and high-speed VLSI design with low supply voltage through cooperation between levels", International Symposium on Quality Electronic Design, 2002. Proceedings. 18-21 March 2002 Page(s):445 – 450.ch2-12

[50] V. Tiwari, S. Malik, and P. Ashar, "Compilation techniques for low energy: an overview," In Proc. of 1994 IEEE Symp. On Low Power Electronics, San Diego, California, USA, Oct., 1994, pp. 38–39.

[51] J. S. Wang, C. N. Kuo, and T. H. Yang, "Low power fixed-width array multipliers," in Proc. IEEE Symp. Low Power Electron. Des., 2004, pp. 307–312.

[52] I. J. Choi, J. Jeon, and K. Choi, "Power minimization of functional units by Partially guarded computation," in Proc. IEEE . Int. Symp. Low Power Electron Des 2000, pp. 131–136.

[53] O. Chen, R. Sheen, and S. Wang, "A low-power adder operating on effective dynamic data ranges", IEEE Trans. Very Large Scale Integration (VLSI) Syst., vol. 10, no. 4, pp. 435–453, Aug. 2002.

[54] O. Chen, S.Wang, and Y. W.Wu, "Minimization of switching activities of partial products for designing low-power multipliers," IEEE Trans.Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 3, pp. 418–433, Jun. 2003.

[55] L. Benini, G. D. Micheli, A. Macii, E. Macii, M. Poncino, and R. Scarsi, "Glitching power minimization by selective gate freezing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 8, no. 3, pp. 287–297, June 2000.

[56] M. C.Wen, S. J.Wang;, and Y. N. Lin, "Low-power parallel multiplier with column bypassing," Electron. Lett., vol. 41, no. 12, pp. 581–583, May 2005.

[57] W. C. Yeh and C. W. Jen, "High-speed Booth encoded parallel multiplier design," IEEE Trans. Comput., vol. 49, no. 7, pp. 692–701, Jul.2000.

[58] N. N. Reddy, J. K. Das, K. K. Mahapatra " FPGA Implementation of an Adaptive Hearing Aid algorithm using Booth- Wallace multiplier", Internation Conference on VLSI Design and Embedded Sytems(ICVLSI'08), pp. 198-202, 14^{th} – 16^{th} Feb, 2008.

[59] D. Hennan, E. Chau, R.D, Dony, S.M. Areibi "Window Based Prototype filter design for highly oversampled filterbanks in audio applications", Proceedings of IEEE ICASSP ,Honolulu ,April 2007.

[60] P. M. Aziz, H. V. Sorenson and J. Van der Spiegel, "An Overvi-ew of Sigma Delta Converters," IEEE Signal Processing Maga-zine, vol. 13, pp. 61-84, January 1996.

[61] B. Leung, "The oversampling technique for analog to Digital conversion : A Tutorial Overview", Analog Integrated Circuits and Signal Processing, Kluwer Academic Publisher, 1991, pp. 65-74.

[62] J. C. Candy, "Decimation for sigma-delta modulation," IEEE Trans. Communication, vol. COM-34, Jan. 1986, pp. 72–76.

[63] L. Naviner, J. F. Naviner, “Design and Prototyping of a Σ Δ Decimator Filter for DECT Standard”, Proceedings of 43rd IEEE Midwest Symposium on Circuits and Systems, 2000.

[64] Shailesh Nerurkar, Khalid. H. Abed, Raymond E. Siferdand, Vivek Venugopal, “Low power Sigma-delta Decimation Filter”, Proceedings of 45th IEEE International Midwest Symposium on Circuits and Systems, 2002.

[65] Stanley A. White, “Applications of Distributed Arithmetic to Digital Signal Processing: A Tutorial Review” IEEE ASSP Magazine July, 1989.

[66] Xilinx Technical article on “The Role of Distributed Arithmetic in FPGA-based Signal Processing”, pp. 1-15, www.xilinx.com.

[67] P. P. Vaidyanathan and Truong Q. Nguyen, “A “TRICK” for the Design of FIR Half-Band Filters” IEEE Transactions On Circuits And Systems, Vol. 34, No. 3, March 1987 pp. 297-300.

[68] Xi Zhang, Kentaro Inotsume, and Toshinori Yoshikawa, “Design of low-Delay FIR Half-Band Filters with Arbitrary Flatness and Its Application to Filter Banks”, Electronics and Communications in Japan, Part 3, Vol. 83, No. 10, 2000, pp. 1-9.

[69] V. Oppenheim and R. W. Schafer, Discrete-Time Signal Processing, Prentice-Hall, 1989.

[70] Steven W. Smith, “The Scientist and Engineer's Guide to Digital Signal Processing”, Chapter 22, Second edition, 1999.

[71] P.S.Sathidevi and Y. Venkataramani, "Enhancement and Modification of Speech and Audio for the Hearing Impaired", Journal of Acoustical Society of India, Vol.30, October, 2002.

[72] S. Ramamoha, S. Dandapat "Sinusoidal Model based Analysis and Classification of Stressed Speech", IEEE Trans. On Speech, Audio and Language Processing, Volume: 14, Issue 3, pp- 737- 746, May, 2006.

[73] R. Dhiman, A. Kumar and R. Bahl, “Design of low power multi-DSP module for acoustic signal processing on remote platforms,” Proc. Intl. Conf. on Sonar – Sensors and Systems, Kochi, pp.361-370, Dec. 2002.

[74] A. Kumar and S. K. Mullick, "Nonlinear dynamical analysis of speech," Journal of Acoustical Society of America, Vol. 100, No. 1, pp. 615-629, 1996.

[75] V. Venugopal, K.H. Abed and S.B. Nerurkar "Design and implementation of a Decimation Filter for hearing aid applications", Proceedings of IEEE South-east Con, April 2005, pp. 111-115.

[76] Edwards,. "Signal Processing Techniques for a DSP Hearing Aid", IEEE International Symposium on Circuits and Systems, June 1998.

[77] Othman O.Khalifa, M.H Makhtar, and M.S. Baharom, "Hearing Aids System for Impaired People", International journal of computing and Information science,vol.2,no.1, pp 23-26,2004.

[78] F. Buergin, F. Carbognani and M. Hediger, "Low Power Architecture Trade-offs in a VLSI Implementation of an Adaptive Hearing Aid Algorithm", IEEE, pages 558- 561, July 2006.

[79] Salina Abdul Samad, Ali O. Abid Noor and Aini Hussain, "An Approach for Low Power Hearing Aids Design", 2008 International Conference on Electronic Design, December 1-3, 2008, Penang, Malaysia.

[80] A. Schaub and P. Straub. Spectral sharpening for speech enhancement/noise reduction. In IEEE Acoustics, Speech and Signal Processing (ICASSP-91), pages 993–996, April 1991.

[81] Wallace, C.S., "A suggestion for a fast multiplier", IEEE Trans. Electron. Compute. vol. EC-13, pp. 14-17, 1964.

[82] A. D. Booth, "A signed binary multiplication technique", Quart. J. Math., vol. IV, pt. 2, 1951.

[83] L. P. Rubinfield, "A proof of the modified booth's algorithm for multiplication," IEEE Trans. Computers, vol. 37, 1988.

[84] M.J.Liao, C.F.Su, Chang and Allen Wu "A carry select adder optimization technique for high-performance Booth-encoded Wallace tree multipliers" IEEE International Symposium on Circuits and Systems 2002.

[85] J. Wassner, H. Kaeslin, N. Felber, and W. Fichtner. Waveform coding for low-power digital filtering of speech data. *IEEE Transactions on Signal Processing*, 51(6):1656– 1661, June 2003. 68

[86] Regalia, Phillip A., Adaptive IIR Filtering in Signal Processing and Control, Marcel Dekker, Inc., 1995.

[87] M. Nayeri and W. Jenkins, 1989, "Alternate realizations of adaptive IIR filters and properties of their performance surfaces", IEEE Transactions on Circuits and Systems, Vol. 36, pp. 485-496.

[88] "Digital system design using VHDL", Charles H Roth, Jr. Thomson Brooks/Cole.

[89] J. A. Maxwell and P. M. Zurek, "Reducing acoustic feedback in hearing aids," IEEE Trans. Speech Audio Processing, vol. 3, pp. 304–313, July 1993.

[90] Sankarayya, N., Roy, K., and Bhattacharya, D. "Algorithms for Low-Power and High-Speed FIR Filter Realization Using Differential Coefficients". IEEE Trans. On Circuits and Systems, Vol. 44, No. 6, pp. 488-497, Jun. 1997.

[91] Simon Haykin, "Adaptive filter", PHI Publication

[92] http://www.xilinx.com /

[93] Flavio Carbognani, Felix Buergin, Norbert Felber, Hubert Kaeslin and Wolfgang Fichtner "42% Power Savings through Glitch Reducing Clocking Strategy in a Hearing Aid Application", Proceedings of IEEE International conference on Circuits and Systems (ISCAS), 2006. pp 2941-2944.

[94] N. Nedovic and V. G. Oklobdzija, "Dual-edge triggered storage elements and clocking strategy for low-power systems", IEEE Transactions on VLSI Systems, May 2005, Vol. 13, No. 5, pp. 577-590.

[95] T. A. Johnson and I. S. Kourtev, "A single latch, high speed double-edge triggered flip-flop (DETFF)", in Proc. The 8th IEEE International Conference on Electronics, Circuits and Systems (ICECS 2001), pp. 189-192.

[96] V. Zyuban, "Optimization of scannable latches for low energy," IEEE Transactions on VLSI Systems , Oct. 2003, Vol. 11, No. 5, pp. 778-788.

[97] P. Mosch, G. van Oerle, S. Menzl, N. Rougnon-Glasson, K. van Nieuwen Hove, and M. Wezelenburg, "A 660-μW 50-Mops 1-V DSP for a hearing aid chip set," IEEE Journal of Solid-State Circuits, Nov 2000, Vol. 35, No. 11I, pp. 1705-1712

[98] F. Carbognani, F. Buergin, N. Felber, H. Kaeslin, and W. Fichtner, "Two-phase clocking and a new latch design for low-power portable applications," in Proc. Power and Timing Modeling, Optimization and Simulation (PATMOS'05), Leuven, Belgium, Sept. 2005, pp. 446-455.

[99] Liu Po Ching and Ong Geok Ling, "Low-power and low-voltage D-latch" IEEE Electronics Letters, Apr. 1998, Vol. 34**,** No. 7, pp. 641-642.

[100] V. Zyuban, D. Meltzer, "Clocking Strategies and Scannable Latches for Low Power Applications", International Symposium on Low Power Electronics and Design, 2001, pp. 346-351.

[101] Arm, C., Masgonty, J., Piguet, C. Double-Latch Clocking Scheme for Low-Power I.P. Cores", Proceedings of the 10th International Workshop on Inte-

grated Circuit Design, Power and Timing Modeling, Optimization and Simulation, 2000, pp. 217-224.

[102] Mahesh Mehendale, Sunil D. Sherlekar and G.Venkatesh" Low Power Realization of FIR Filters on Programmable DSPs", IEEE Transactions on VLSI Systems, Dec 1998, Vol.6, No.4, pp.546 – 553.

[103] Volnei A Pedroni, "Circuit Design with VHDL", PHI Publication.

[104] The National Technology Roadmap for Semiconductors, Semiconductor Industry Association (SIA), 1999–2000.

[105] C. R. Xiong, J. F. Hu and Y. K. Hou, "Research and Design Based on DSP System of SOPC," Microcomputer Information, Vol. 24, No. 6-2, pp. 206-208, June 2008.

[106] A. Jain et al., "A 1.2 GHz alpha microprocessor with 44.8 GB/s chip pin bandwidth," in IEEE Int. Solid-State Circuits Conference Technical Digest, Feb. 2001, pp. 240–241.

[107] P. Hofstee *et al.*, "A 1-GHz single-issue 64b powerPC processor," IEEE International Solid-State Circuits Conference, Feb. 2000, Vol. 33, No. 11, pp. 92-93.

[108] R. P. Llopis and M. Sachdev, "Low power, testable dual edge triggered flip-flops," International Symposium on Low Power Electronics and Design, 1996, pp. 341–345.

[109] A. Gago, R. Escano, and J. A. Hidalgo, "Reduced implementation of D-type DET flip-flops," IEEE Journal of Solid-State Circuits, Mar. 1993,Vol. 28, No. 3, pp. 400–402.

[110] J. Tschanz, S. Narendra, Z. Chen, S. Borkar, M. Sachdev, and V. De, "Comparative delay and energy of single edge-triggered & dual edgetriggered pulsed flip-flops for high-performance microprocessors," International Symposium on Low Power Electronics and Design, Aug. 2001, pp. 147–152.

[111] N. Nedovic, M. Aleksic, and V. G. Oklobdzija, "Conditional pre-charge techniques for power-efficient dual-edge clocking", International Symposium on Low Power Electronics and Design, Aug. 2002, pp. 56–59.

[112] N. Nedovic, V. G. Oklobdzija, M. Aleksic, and W. W. Walker, "A low power symmetrically pulsed dual edge-triggered flip-flop," Proceedings of 28th European Solid-State Circuits Conference, Sept. 2002, pp. 399–402.

[113] N. Nedovic and V. G. Oklobdzija, "Timing characterization of dual-edge triggered flip-flops," Proceedings of the International Conference on Computer Design, Sept. 2001, pp. 538–541.

[114] V. Stojanovic and V. G. Oklobdzija, "Comparative analysis of master-slave latches and flip-flops for high-performance and low-power systems", IEEE Journal of Solid-State Circuits, Apr. 1999,vol. 34, no. 4, pp. 536–548.

[115] N. Nedovic, W. W. Walker, and V. G. Oklobdzija, "A test circuit for measurement of clocked storage element characteristics," IEEE Journal of Solid-State Circuits, Aug. 2004, Vol. 39, No. 8,pp. 1294-1304.

[116] D. Markovic, B. Nikolic, and R. W. Brodersen, "Analysis and design of low-energy flip-flops", International Symposium on Low Power Electronics and Design, Aug. 2001, pp. 52–55.

[117] E. P. Zwyssig, A. T. Erdogan and T. Arlsan "Low Power System on Chip Scheme of Digital Filtering cores", IEE Low Power IC Design Seminar, January 2001, London, UK, pp. 5/1-5/9.

[118] F. Carbognani, F. Bürgin, L. Henzen, H. Koch, H. Magdassian, C. Pedretti, H. Kaeslin, N. Felber, W. Fichtner, "A 0.67-mm2 45-µW DSP VLSI Implementation of an Adaptive Directional Microphone for Hearing Aids", European Conference on Circuit Theory and Design (ECCTD), Cork, Ireland, 29. Aug - 1. Sep 2005.

[119] Christian Piguet , "Low-Power CMOS Circuits: Technology, Logic Design and CAD Tools", CRC Press, 2006

[120] A. Chadrakasan and R. W. Brodersen, "Low Power Digital CMOS Design, Kluwer Academic Publisher, 1995.

[121] G. K Yeap, "Practical Low Power Digital VLSI Design", Kluwer Academic Publishers, Norwell, Mass., 1998.

[122] Jan M. Rabey, Ananth Chandrakasan and Borivoje Nikolic, "Digital Integrated Circuits A design Perspective", Pearson Education , 2nd Edition, 2003.

[123] P. Zhao and Z. Wang, "Low Power VLSI Circuits and Systems", IEEE 8th International Conference on ASIC, ASICON'09, 20-23 Oct, 2009, pg. 17-20.

[124] Sung-mo-Kang and Yusuf Leblebici, "CMOS Digital Integrated Circuits", TATA McGraw Hill India, 2nd edition, 2003.

Contribution by the candidate

[P1] N. N. Reddy, J. K. Das, K. K. Mahapatra," FPGA Implementation of an Adaptive Hearing Aid Algorithm using Booth-Wallace Multiplier", International Conference on VLSI Design and Embedded Systems(ICVLSI'08), Velammal Engineering College, Chennai, pp. 198-202, 14^{th} -16^{th} Feb, 2008.

[P2] J. K. Das, K. K. Mahapatra, "Decimation filter for Hearing Aid Application using Distributed Arithmetic", IEE Sponsored Conference on Computational Intelligence, Control And Computer Vision in Robotics and Automation, National Institute of Technology, Rourkela pp 170-174, 10^{th} -11^{th} March, 2008.

[P3] J. K. Das, K. K. Mahapatra, "Design of a Novel Low Power Multiplier using Spurious Power Suppression Technique", International Conference on Communication, Computers, and Instrumentation-2009 from January 2-3,2009, pp-229-234.

[P4] J. K. Das, K. K. Mahapatra, "Low Power Filter Design Using a Novel Dual Edge Triggered Latch", IEEE International Conference on Electronic Design 2008, December 1-3, 2008, Penang, Malaysia.

[P5] J. K. Das & K K. Mahapatra "Design of an Adaptive Hearing Aid Algorithm using Booth-Wallace Tree Multiplier", International Journal of Logic and Computation (IJLP), Volume (1): Issue (1), pp:1-17, September, 2010.