Distributed Arithmetic (DA) Architectures for Signal Processing Applications

By
Minaam Ahmad Naeem
2010-NUST-MS Phd-ComE-13
MS-65 Computer Engineering

Submitted to the Department of Computer Engineering
In partial fulfillment of the requirements for the degree of

Master of Science
In
Computer Engineering

Advisor
Dr. Shoab Ahmed Khan

College of Electrical & Mechanical Engineering
National University of Sciences and Technology
2013

In The name of ALLAH,

The most Beneficent and

The most Merciful

DECLARATION

I hereby declare that I have developed this thesis entirely on the basis of my personal efforts under the sincere guidance of my supervisor Brig. Dr. Shoab Ahmed Khan. All the sources used in this thesis have been cited and the contents of this thesis have not been plagiarized. No portion of work presented in thesis has been submitted in the support of any application for any other degree of qualification to this or any other university or institute of learning

_____

Minaam Ahmad Naeem

DEDICATIONS

Dedicated to My Mother, My Father, My Wife, My Sisters, My Teachers,

Family and Friends

ACKNOWLEDGEMENTS

All praises be to **Allah Almighty**, The most Gracious and The most Merciful for the endless blessings bestowed upon me and for the courage and consistency provided by Him to complete this task throughout this tough period.

First of all an expression of special and deep gratitude to my supervisor **Brig. Dr. Shoab Ahmed Khan**, without whose guidance, motivation and endless support, it could never had become possible to remain sustainable throughout this thesis work. Thank you very much Sir for the immense motivation and support provided by you.

I would like to thank and express my deep gratitude to **Dr. Almas Anjum, Dr. Khalid, Dr. Saad Rehman, Dr. Arslan Shaukat, Dr, Usman Qamar, Dr, Sheikh M. Farhan, Dr. Farhan Riaz. Dr. Usman Akram** and all the teachers whose imparted knowledge and guidance made me capable enough to carry on and complete this thesis work.

The support I got from **My Parents, My Wife, My Sisters and Family** and without their prayers and encouragement nothing can be possible. Thank you very much everyone. Thanks to all my Friends and colleagues whose support facilitated me a lot.

ABSTRACT

The applications of today's high tech digital world operate on ultra-fast processing speeds and their hunger for more resource is increasing. Especially in signal processing and communication systems precision and accuracy are equally important besides the giga hertz frequencies. Implementing algorithms by using alternate architectures that require least amount of resources as well as minimum delays has become a must.

FIR filter being the most widely used filter in digital processing applications is a non-ideal filter whose high orders need to be used in architecture for accuracy. Multiplication is one of the most resource consuming and time consuming operation in any computation and is the basic part of FIR filters.

Digital systems usually involve multiplication of constant coefficients with variable data input. Distributed arithmetic (DA) architecture exploits this fact and provides a multiplier less multiplication solution to the most computational complex operation. It has been in use with various modifications for past couple of decades.

The main idea functional behind the distributed arithmetic (DA) architecture is to dig deep up-to bit-level and pre-compute the possible answer to the multiplication prior to the arrival of the input bits. These pre-computed values are saved as ROMs and embedded in hardware. This makes multiplication operation just a matter of selection of correct answer among available values.

The distributed arithmetic (DA) architecture has been studied for resource utilization by designing and implementing FIR Filters in Verilog and synthesizing for Viretex5 FPGA. Same idea is also used in single carrier as well as multi carrier digital communication system to study optimization in resource utilization. Further improvements have also been proposed. Most research is focused on efficient use of resources on FPGA through implementation.

Table of Contents

CHAPTER 1

INTRODUCTION

CHAPTER 2

DIGITAL SIGNAL PROCESSING

CHAPTER 3

LITERATURE REVIEW

CHAPTER 4

IMPLEMENTATIONS AND MODIFICATIONS

CHAPTER 5

RESOURCE UTILIZATION ANALYSIS

CHAPTER 6

CONCLUSION AND FUTURE WORK

List of Figures

Chapter 1

Introduction

This chapter introduces the carried research work along with defining the problem statement and the methods used for solution. Basic concepts are also discussed for a brief understanding and comprehensive revision.

1.1 Motivation

Since the beginning of modern age, data communication and computation has been one of the most revolutionary fields and history is full of incredible breakthroughs. Invention of digital computer has proved to be a doorstep towards a complete new era of science and technology. Various new dimensions have been introduced with by concept of VLSI (Very Large Scale Integration) and has made it is possible to solve the most challenging questions regarding any field known to humans. Digital communication systems, multimedia and broadband applications, biotechnology, robotics, computer networking, mobile communication and data processing storage & security have been the latest fields that fully utilize this computational power and still the hunger for more is on the rise.

Starting from abacus, hardware platforms capable of supporting applications in mentioned fields has evolved as Pentium processors, General Purpose Processors (GPPs), Digital Signal Processors (DSPs), Application Specific Integrated Circuits (ASICS), Field Programmable Gate Arrays (FPGA), Network on Chip (NoC) and System on Chip (SoC).

FPGA has emerged as one of the most powerful hardware equally suitable to design, test and implement all types of architectures which involve super-fast processing and high frequencies. Ultra-fast mesh of paths is connecting all the resources like multipliers, adders, look-up-tables, shifters, memory and registers to high end processor making it possible to run all the desired architectures and techniques for multiple purposes. Due to the limitations including price, power consumption, processing speeds, transferring speeds etc. there are only few resources available on any single FPGA platform which have to be managed very efficiently to successfully accomplish the task. The on board resources are very scarce which are required for evolving complicated applications in digital communication, image processing and multimedia, satellite transmissions and medical computerized instrumentation. The speed factor is a very crucial parameter for such applications.

This takes the task to a complete next level; not only such new architectures and techniques have to be designed and tested for accuracy, but also these have to be implemented in an optimized way to minimize the use of already scarce available resources. A user can always find an FPGA having all the required huge amount of resources on a board, but might not be able to pay for the same and here the fight for the most resource efficient architecture starts. Ever since, people have dedicated their studies and research in devising ways and implementing shortcuts which would help everyone in resolving the issues of efficient resource utilization. This has become a vast field of study and now is a must so as to minimize the price if product has to be introduced in the market.

In the field of communication, specifically, users would always like a very high speed, the faster the better. This requires a high speed and very efficient architecture using high end processing device and manipulating with all the resources very efficiently so that least resources are consumed and could possibly be made available for other purposes if required.

Architectures are under a lot of attention to cope with this drought of resources by introducing alternate methods for high resource consuming operations like multiplication. Distributed Arithmetic (DA) is one such architecture that successfully eliminates the multiplication operation and provides a solution that is fast, and less costly in terms of utilized resources.

1.2 Background

Resources available at hand need to be utilized very efficiently for minimum cost and maximum output. Different architectures are being developed that provide alternate methods for resolving this issue. FIR filters are widely used in digital applications relating to communication and signal processing like equalizers, receiving filters, estimation, image filtering and signal reshaping. Its wide use has made it necessary to design and implement algorithms to get the same results by using alternate methods without compromising on accuracy and precision and also use least amount of resources in its multiplication operation.

Distributed Arithmetic (DA) architecture has been very successful is solving the issues. Its design was introduced by A.Croisier[1] and S.A. White [2]. The idea has further been polished by H. Yoo and D. V. Anderson [3], P. Longa and A. Miri [4], W. Sen, T. Ben and Z. Jim [5]. Implementation for single carrier communication system is based on design introduced by Park, Richards and Borivoje [6]. For OFDM system, scheme introduced by Selimane [7] is followed and test case is implemented to study the results. Dr. Shoab Ahmed Khan [8] in “Digital Design of Signal

Processing Systems; A Practical Approach" has explained the architectural design and actual working of different DA based techniques along with practical examples and RTL diagrams.

1.3 Problem Statement

The Problem Statement States that there should be a method which minimizes the utilization of very expensive resource on the FPGA. This should also increase the maximum operating frequency of the system. FIR filter involves multiplication operation that consumes time as well as huge resources and is used as base for our study, design, implementation and testing purpose.

1.3.1 Distributed Arithmetic (DA) Architecture

FIR filter is implemented using Distributed Arithmetic (DA) architecture. Further modification is introduced for better results. The results are compared with the conventional implementation of FIR filter for accuracy as well as performance and resource utilization.

1.3.2 Equalizers in single carrier communication system

The modified design is used to implement equalizers at receiving end of single carrier communication working on BPSK modulation.

1.3.3 Precoding in OFDM multicarrier communication system

The Distributed Arithmetic (DA) architecture is also designed, implemented and tested for multicarrier OFDM communication system. It implements the most resource consuming very high order multiplication operation used at both transmitter and receiver sides during precoding algorithm.

1.4 Problem Solution

The aim of this research work is to implement and analyze an alternate architecture to minimize the utilization costly resources on FPGA. The modified design in combination to the existing architectures successfully addresses this issue and analysis of results reveal the advantage gained in signal processing involved in single carrier as well as multicarrier systems.

1.5 Outlines of thesis

Thesis is compiled in following manner

- Chapter 2: Digital Signal Processing

  In this chapter, basic concepts related to Digital Signal Processing are described. Both time domain as well as frequency domain techniques are discussed. The nature and working of FIR filter and the implementation are discussed. A typical single carrier communication system is discussed with focus on the equalizers used at receiver. Components and concepts of multi carrier OFDM communication system are reproduced and precoding method is elaborated to counter the PSPR issue.

- Chapter 3: Literature Review

  Chapter 3 provides the history of developments made in architecture design. The discussions are made on published works and main themes are discussed.

- Chapter 4: Implementations and Modifications

  The solution architecture is designed and implemented along with few modifications for enhanced performance

- Chapter 5: Resource Utilization Analysis

  The utilization of resources are discussed as a comparison to describe achieve results and final conclusion is made.

Chapter 2

Digital Signal Processing

Digital signal processing (DSP) is a vast field which is concerned to process the digital signals relating to any domain so that the signal can be analyzed or the data be extracted or used as per requirement. The most powerful and reliable tool used for accomplishing the task is mathematics. Symbols represent signals and numbers are weights of coefficients. In modern times, DSP techniques have found its use in almost all the modern world applications of science and technology. Few major fields are

- Communication Systems
- Robotics
- Control Systems
- Biotechnology
- Data and Statistical Processing
- Digital Image Processing
- Audio and Speech Signal Processing

Special powerful Digital Signal Processors (DSPs), Application Specific Integrated Circuits (ASICS), General Purpose Processors (GPPs) and Field Programmable Gate Arrays (FPGAs) are now in vast production and wide range of these platforms is available for implementation and realization of DSP architectures functional at super-fast speeds.

2.1 Time Domain Analysis

In Time Domain, all the signals are dealt as time samples of signals and further filtering is applied on these signals. Almost all the real world signals are already in time domain. Main types are

- **Linear Filter**: The functionality of such filters is linear i.e. output is only a weighted version of the input signal.
- **Causal Filter**: It uses samples from the past to compute the output.
- **Time Invariant Filter**: The functionality of these filters never change with time.
- **Stable Filter**: The output of such filter must converge to a constant value if extended to some point in time.
- **Finite Impulse Response (FIR) Filter**: An always stable filter only considering finite input sample values to compute output.
- **Infinite Impulse Response (IIR) Filter**: It consumes both input and old output sample values (as feedback) to determine new output.

2.2 Frequency Domain Analysis

In frequency domain, signal under analysis is in the form of phase & magnitude. Fourier transform converts the real world time domain signal into frequency domain signal. Frequency and phase information present in the signal can only be determined in frequency domain. The relationship between frequency and phase variation is of great importance in communication systems.

2.3 Finite Impulse Response (FIR) Filter

In DSP, especially in communication systems, the most widely used filter is Finite Impulse Response (FIR) Filter. It uses only finite number of input samples to compute the output and the impulse response is also of finite duration. FIR filter of order N (i.e. having "N" number of coefficients), the response to delta (δ) impulse signal input gives N+1 samples in before it sets to zero. It is equally useful in discrete/continuous time and digital/analog domains.

The output of FIR Filter is a weighted (with respect to Filter coefficients) sum of current sample and few finite previous values of input samples. Mathematically,

$$\mathrm{y}[n] = b_0 x[n] + b_1 x[n-1] + \cdots + b_N x[n-N] \quad (2.1)$$

$$\text{Or} \quad y[n] = \sum_{i=0}^{N} b_i \, x[n-i] \quad (2.2)$$

In above equation (2.2):

- $x[n]$ is input signal to FIR filter
- $y[n]$ is output signal from FIR filter
- b_i are the FIR coefficients
- N is the order of FIR filter

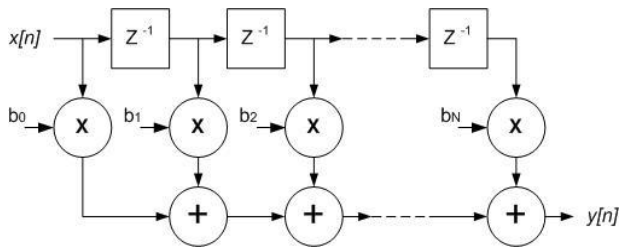


Figure 2.1 Finite Impulse Response (FIR) filter

2.3.1 Impulse response of FIR filter

The Impulse response of FIR filter can be determined by setting the delta function as the input to FIR and recording the output. The result is simply the discrete set of coefficients in the FIR filter. Mathematically it is represented as

$$h[n] = \sum_{i=0}^{N} b_i \, \delta[n-i] = b_n \tag{2.3}$$

In above equation (2.3):

- $\delta[n]$ is the input delta impulse signal to FIR filter
- $h[n]$ is the output impulse response from FIR filter
- b_i , b_n are the set of coefficients of FIR filter

2.3.2 Transfer function of FIR filter

Transfer function can be obtained by taking the Z-Transform of Impulse of FIR Filter as in Equation 2.3. Mathematically it is represented by

$$H(z) = Z\{h[n]\} \tag{2.4}$$

$$H(z) = \sum_{n=-\infty}^{\infty} h[n] \, z^{-n} \tag{2.5}$$

$$H(z) = \sum_{n=0}^{N} b_n \, z^{-n} \tag{2.6}$$

2.3.3 Implementation of FIR Filter

As clear from block diagram (figure 2.1) and mathematical representation (equation 2.2), FIR is implemented as a multiplier and accumulator (MAC) operation. This is the biggest motivation for deployment of dedicated MAC units on DSP and FPGA kits e.g. DSP48e block is a MAC unit with multiplier and adder and register for temporary storage. Steps involved are described below.

2.3.3.1 Multiplication

All the input samples are multiplied with their corresponding FIR Filter coefficients. Block diagram illustrates the operation.

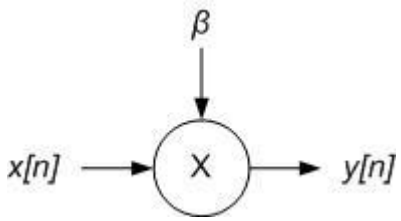


Figure 2.2: Multiplication operation in FIR

Mathematically

$$y[n] = \beta x[n] \tag{2.7}$$

For implementation, the multiplier used in DSP needs to be quick and precise to support this operation as the signals are in Mega and Giga Hertz. For high precision, the order of FIR filter is kept high to approach nearest to ideal condition. This means that the number of multiplications required in FIR filters is large and both hardware as well as architecture used must support such high speed computation.

2.3.3.2 Addition

These individual sample-coefficient multiplication results are then added up. So an accumulator is also required. This is also a very basic operation in DSP systems. Block diagram depicts the operation.

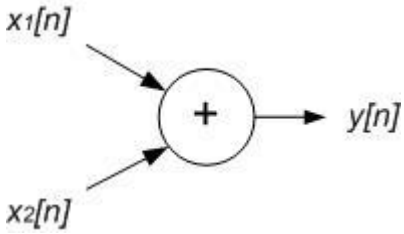


Figure 2.3: Addition Operation in FIR

Mathematically

$$y[n] = x_1[n] + x_2[n] \tag{2.8}$$

As the number of additions is one less than the number of multiplications, so adders must also be fast and precise. The main issue with the adders it is a sequential operation i.e. only two bits can be added at a time and the carry must also be forwarded to next bit. This increases the computational cost and the resources required for effective addition.

2.3.3.3 Unit Delay

If we consider the above mentioned MAC operation as a stage, then each input sample stays at a stage for one clock cycle. On the next clock trigger, this input sample moves to the next stage where it goes through MAC again with different coefficient. So every next stage is computing output on input sample delayed by one unit from its previous stage and so on. Block diagram depicts the operation.

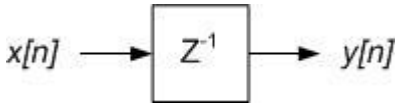


Figure 2.4: Unit Delay Operation in FIR

Mathematically

$$y[n] = x[n-1] \tag{2.9}$$

This means that for N number of unit delays we need enough storage space in registers to save N input samples having width of K bits (K depends on number of quantization levels or precision required) depending on the digital system requirement.

2.3.4 Properties and Importance of FIR Filter

There are many useful properties of FIR filters which are exploited in all digital systems for the benefit. Few major ones are described below which show the importance as well.

- Linear phase filters are those which cause equal delay or react equally to all the frequencies in the signal under processing. The response of FIR filter can easily be designed as linear phase just by keeping symmetric the sequence of coefficients. This is highly advantageous in especially in digital communication systems.
- Stability of digital system means that all the poles are inside the unit circle (for Z-Transform). As FIR filters never involve its output values again as input, so FIR filters are stable. This means that output converge to a single finite point. This property also depicts the fact that as there is no feedback involved, the error is constant for all outputs. The error never adds up to produce a snowball effect and destroy the whole system.
- The number samples are finite, making it a good choice for systems involving interpolation or decimation. Also this makes the implementation on DSP processors easier. Also the bit-

precision is not lost as new input enters the system and outputs may be discarded; unlike IIR where the repeated use of same output may result in loss of required bit precision.

2.3.5 Challenges in Implementation of FIR Filter

FIR filter uses only finite number of input samples for computation. Therefore the response is never a perfect square wave as we get in case of ideal filtering or IIR filters. Few samples of FIR filter response are given in the figure by varying the order of the filter. It is evident from the practical example shown in figure 2.5 and figure 2.6 that even after very high number of FIR filter the ideal could not be achieved. In order to get as close to the ideal response, the order of FIR filter has to be increased.

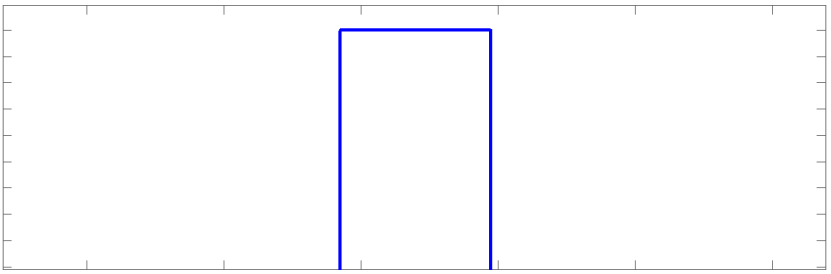
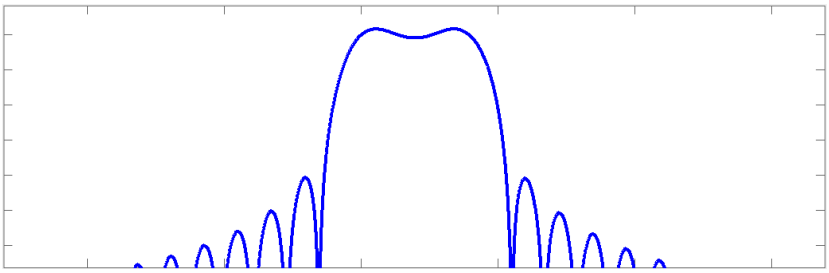
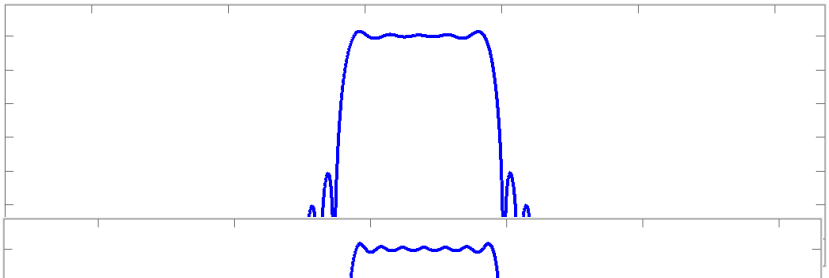


Figure 2.5: Response of an ideal band pass filter

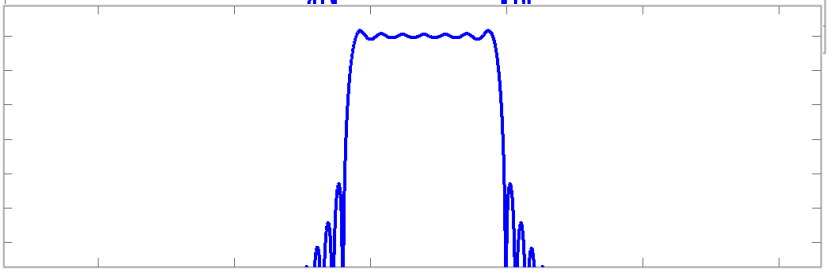
(a) Response of FIR filter with 100 coefficients



(b) Response of FIR filter with 200 coefficients

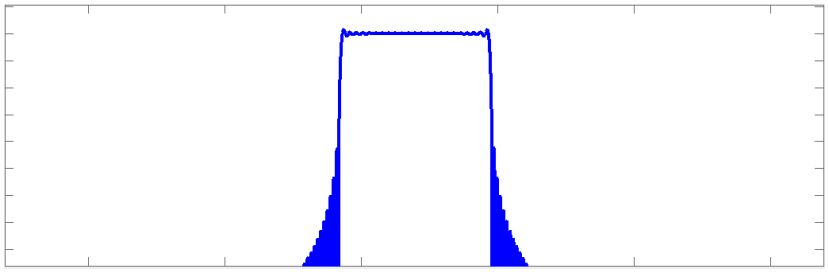


(c) Response of FIR filter with 300 coefficients



(d) Response of FIR filter with 400 coefficients

(e) Response of FIR filter with 1000 coefficients



(f) Response of FIR filter with 2000 coefficients

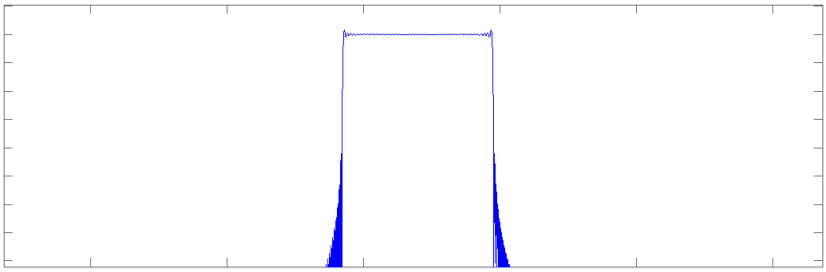
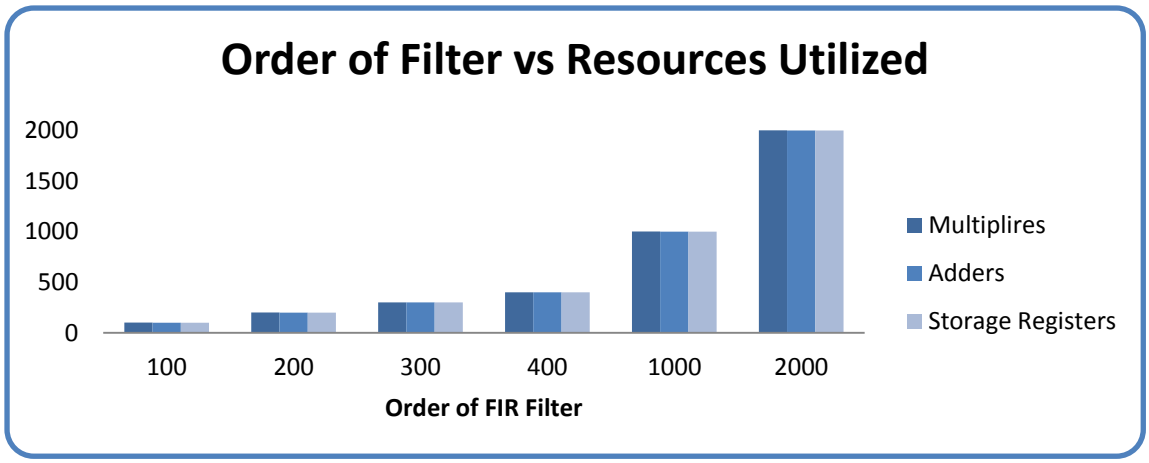


Figure 2.6: Responses of band pass FIR filter

Now, this increase in order of FIR is followed by increase in the number of multipliers, adders and storage registers making this FIR filter a very computationally intensive operation. Table (2.1) shows this fact numerically.

Order of FIR Filter	No. of Multipliers	No. of Additions	No. of Storage Registers (b-bits wide)
100	100	99	100
200	200	199	200
300	300	299	300
400	400	399	400
1000	1000	999	1000
2000	2000	1999	2000

(Table 2.1 Resources required in different order of FIR filter)



The resources on the DSPs, FPGAs are very scarce and expensive. Therefore a trade-off has to be made between the cost of platform for implementation and the precision of results from FIR filters.

Another innovative way is to choose an altogether different architecture to implement these MAC operations. Scientists, engineers and researchers have dedicated their work to explore new ways so that such an exhaustive computation can be replaced by some alternate.

2.4 Single Carrier Communication System

IEEE WPAN standard [9] explains single carrier modulation in details. The single carrier modulation scheme keeps the concept simple by modulating on only one carrier. The receiver end of this system is easily re-configurable. Also, the power required can be altered with respect to the channel's actual live conditions. These inherent properties give the edge to use single carrier modulation technique for ultra-fast communication systems in adaptive mode where we have very limited power budget in hand to play with.

2.4.1 Modulation Scheme

In this research work, BPSK (Binary Phase Shift Keying) modulation scheme is used. It allows understanding and implementing the design in a simpler way. As the figure explains, two constellation points are used which maximizes the separation i.e. 180^{o} allowing maximum tolerance and minimum BER (Bit Error Rate). Only 1 bit represents the BPSK symbol. Few alterations to the design will make possible to use QPSK as well.

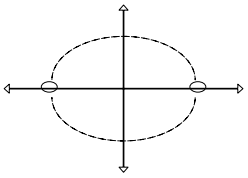


Figure 2.7: BPSK Modulation

2.4.2 Frame Structure

The information to be sent to receiver not only consist the actual data but a lot many other overhead bits and signals which make the communication possible. These provide certain control signals, synchronization information and channel behavior which is vital at the receiver. Information in shape of bits is packed together in blocks called frames. Common frame structure is shown in figure (2.8).

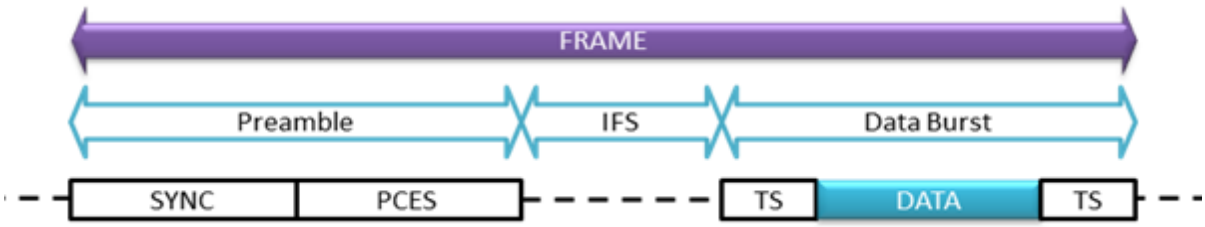


Figure 2.8: Frame Structure

2.4.2.1 Preamble

This part is sent before the actual data burst as it has to estimate parameters before data enters the channel.

- SYNC: These pilot sequences are of short period. These are used for detection, gain control and initially synchronizing the timing as well as frequency. These are required by channel estimator and synchronization estimator.
- PCES: Channel estimator uses preamble channel estimation sequence initially to estimate the channel.

2.4.2.2 IFS (Inter Frame Spacing)

To cater for the latency caused by initial estimators, IFS is used. It is a spacing of few micro seconds which usually equals to thousands of symbols. Parameters that need to be used in coming data bursts are sometimes calculated. This also conserves the power consumed.

2.4.2.3 Data Burst

- TS: These short period segments help in keeping the synchronization among Transmitter as well as receiver; relating to clock, time shift and frequency. This is known as Synchronization Tracking and is used in Channel Estimator as well as Synchronization Estimator.
- DATA: The actual payload or message or data is placed inside this segment. It is processed in Equalizer to get required results.

2.4.3 Receiver Structure

After leaving the transmitter, the signals pass through different channels and are affected in multiple ways i.e. they can experience amplitude / frequency / phase shifts or delays, attenuation, noise and much more. In order to receive the transmitted data accurately, the receiver has to be empowered with the best available algorithms running on high profile hardware. This makes receiver the most complex part of communication system. Main components are shown in figure (2.9)

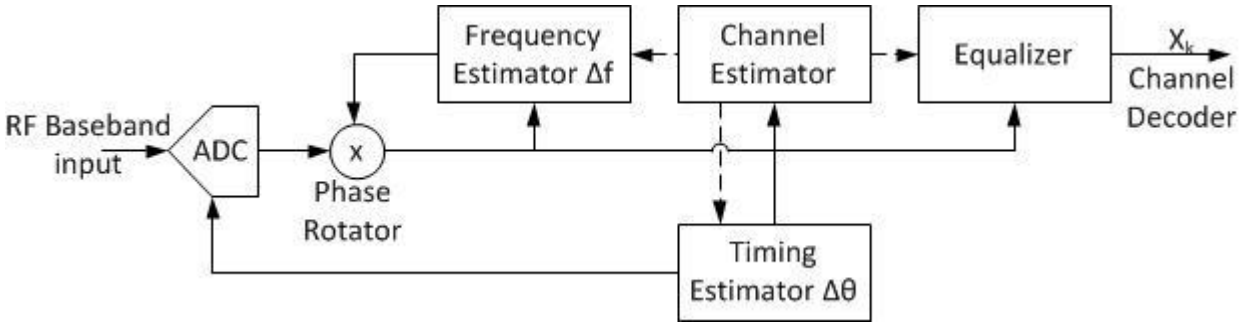


Figure 2.9: Typical Receiver Structure

The input signal is in the form of base band RF analog signal. It is converted into digital signal through analog to digital convertor. The channel estimator receives the digital signal and estimates the channel impulse response of channel on the signal. According to these estimated parameters, instructions are given to equalizer, timing estimator and frequency estimator to take action accordingly and to take measures so that next signals can be processed in a more accurate way. The frequency estimator on receiving the actual signal and certain parameters from channel estimator decides what action on frequency is required. It makes these changes accordingly into phase rotator so that channel effect is cancelled on next receiving signal. Timing estimator on receiving the actual signal as well as the parameters from channel estimator adjusts the timing of the ADC detecting module and so upcoming symbols are not disturbed.

2.4.3.1 Equalizer

Before reaching the receiver, signals pass through different channels. For a channel to be ideal, the delay must be constant for all frequencies in the used spectrum. If the group delay is not constant then every frequency gets separate delays and phase of the signal is disturbed e.g. fading channels. This disturbance in phase as well as amplitude of the signal causes the symbols to disperse

and overlap with other symbols giving a smearing effect known as inter symbol interference ISI. As we increase the transmission speeds, this effect increases and more data gets corrupted. To counter for ISI we use equalizers which use signal processing techniques at the receiver end. There are two main categories of equalizers

2.4.3.1.1 Maximum Likelihood Sequence Estimation (MLSE)

In this type, the actual distorted received signal is not compensated and worked upon to re-shape into the original form. Instead, the channel response is studied and the detector is tuned accordingly so that it can decide the most likely correct data by analyzing same distorted signal. Viterbi equalization works on this principle and has found many applications in communication systems.

2.4.3.1.2 Equalization using Filters

Here, different filters are used to modify the received signal. ISI is removed from the signal and then the decision is taken on this clean signal for detection. This is a more simpler way as we have to process in reverse all the effects created by the channel. Filters can also be manipulated and easily adjusted according to the channel nature. Further, these can be implemented in two different ways

2.4.3.1.2.1 Transversal Equalizers (TE)

These filters work in feed-forward fashion only i.e. they have no feedback system and thus the simplest form of filters. These are also termed as Linear Equalizers (LE). A typical adjustable coefficient LE is shown in figure (2.10). In this only the present and previous values are multiplied linearly with the equalizer weights and summed to get the result.

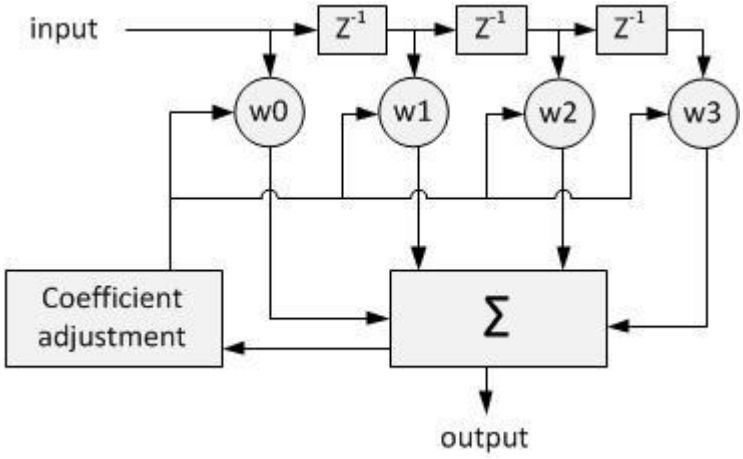


Figure 2.10: Traversal Equalizer

Linear Equalizer taps are usually kept less as they also amplify the noise and also sometimes implementation gets difficult. Its response to the spectral nulls is also not up to the mark which is very common in the mobile and wireless communications.

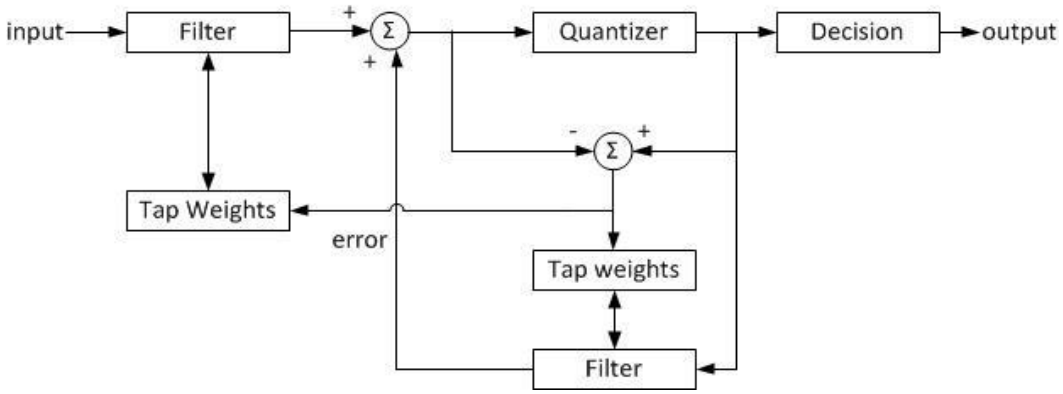


Figure 2.11: Decision Feedback Equalizer

2.4.3.1.2.2 Decision Feedback Equalizers (DFE)

In this type of filters, the decisions taken previously are used to eliminate ISI in the current and coming symbols. The distortions from previous symbols are simply subtracted from the current symbol and so on; this removes the smearing effect very efficiently. Multiple feed-forward filters may also be used in different directions to create a non-linear DFE. The feedback part is in addition to the coefficient adjustment algorithm so ISI is completely removed. A big disadvantage is that once an error is introduced in the system, it can replicate in the loop causing snow ball effect and destroy the effectiveness of decision making.

2.5 Multicarrier OFDM System

Orthogonal Frequency Division Multiplexing is an advanced technique which first splits the signal into multiple orthogonal sub-carriers. Data is modulated on these sub-carriers and then these are multiplexed again which creates OFDM carrier. Inverse Discrete Fourier Transform (IDFT) and Discrete Fourier Transform (DFT) pair is used at transmitter and receiver to realize multiplexing of orthogonal sub-channels. Typical OFDM functional blocks are shown in figure (2.12).

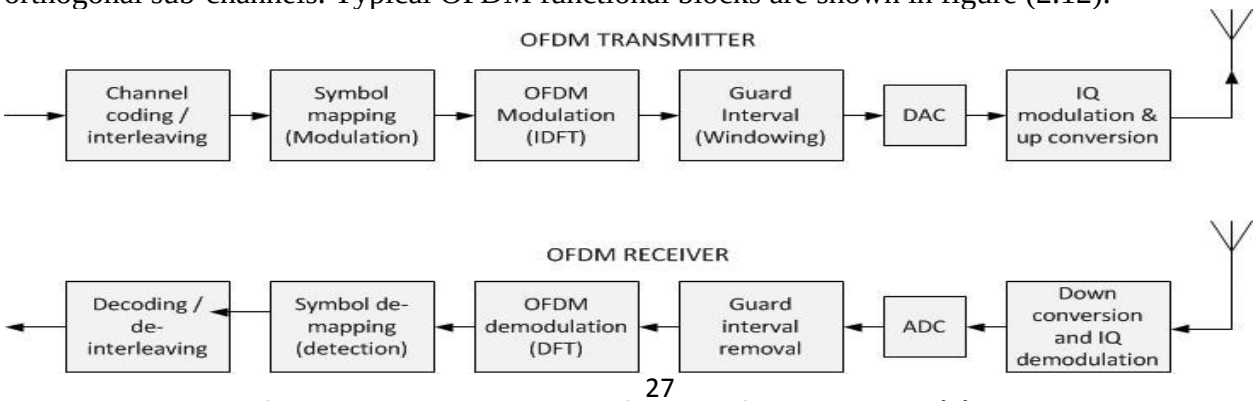


Figure 2.12: OFDM Transmitter Receiver System Model

2.5.1 Working Principles in OFDM

Input to this block is the base band modulated digital data stream in a serial fashion. First block of this is converted from serial to parallel and then passed through IFT at transmitter and vice versa at the receiver. It is inbuilt in the FT and IFT that each parallel bit is multiplied by frequencies orthogonal to each other as discussed in coming section. After this, all these frequency sub-channels are added and resultant is again the serial stream of data. This function is shown in the figure (2.13).

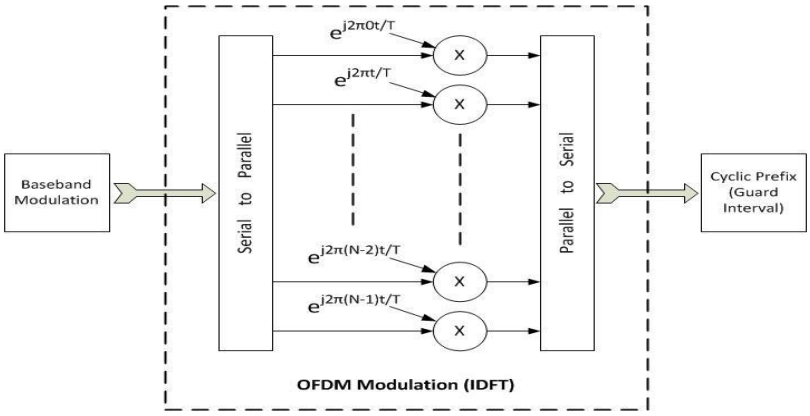


Figure 2.13: OFDM Modulation (IDFT / DFT)

2.5.1.1 Orthogonal nature

In OFDM Systems, the main channel is divided into sub-channels such that these are all orthogonal to each other. For this purpose, FT and IFT pairs are used which split the channels in orthogonal fashion. Only one frequency is a peak and rest all are null. Therefore the product of these orthogonal frequencies will give null hence signal is not disturbed and data is safe.

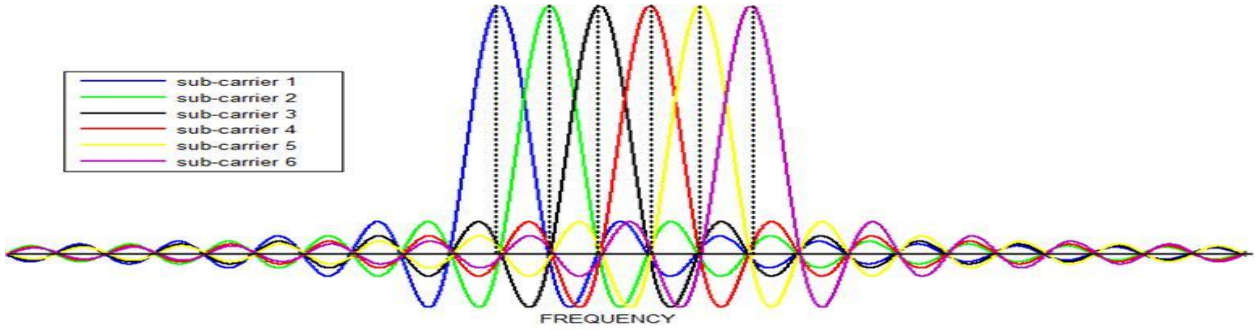


Figure 2.14: Orthogonal Frequency sub-channels in OFDM System

This property ensures high spectral efficiency and no inter-sub-channel interference therefore a very successful technique for high speed communication devices.

2.5.1.2 Guard Interval

To protect the long OFDM symbols from overlapping due to delays, a guard interval is inserted between every consecutive symbol. Small part from the start of previous symbol is copied at the end.

Thus if there is any over lapping then this guard interval is destroyed and the original symbol is kept safe. It keeps safe from Inter Symbol Interference at the cost of Guard Interval Overhead bits. This method is known as cyclic prefix demonstrated in Figure (2.15).

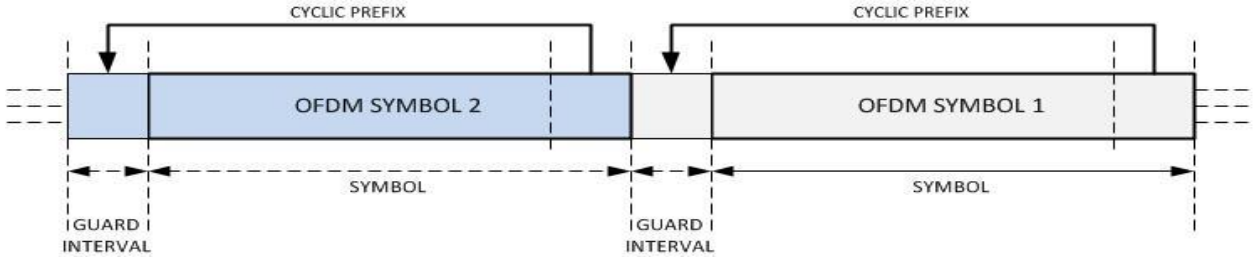


Figure 2.15: Guard Interval placement as Cyclic Prefix

2.5.1.3 Interleaving

In OFDM we have multiple sub-channels and each carrying different data. To avoid the frequency selective fading channels, the bits are put on the sub-channels in the interleaved channels so that concentrated errors don't occur and minimum faults arise. This is done both in frequency domain between the sub-channels and time domain within the symbol.

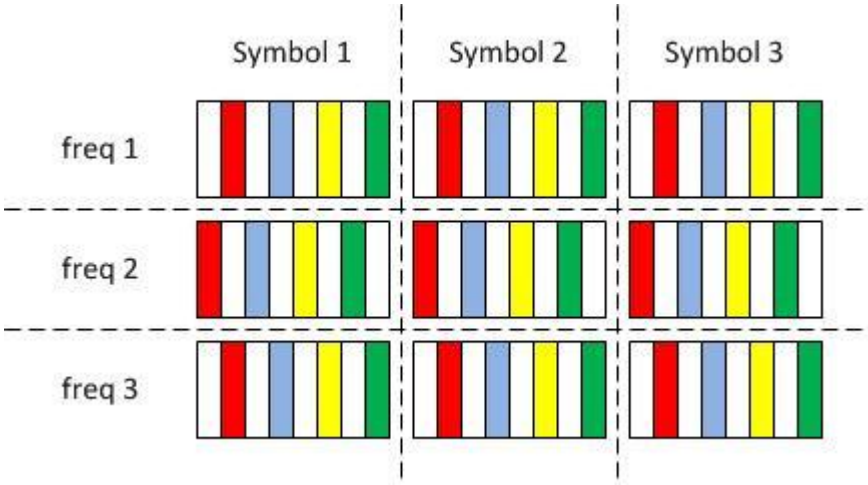


Figure 2.16: Interleaving in OFDM System

2.5.2 Advantages of OFDM System

- **Complexity:** The computational complexity is O(BlogBTm) and implementation is not very complex. FFT/ IFFT pair is used and data rate can easily be varied according to requirements just be inserting more data on interleaved empty slots.

- **Graceful degradation:** As the delay spread increases, performance of an OFDM system degrades gracefully. Greater coding sizes and lower constellation sizes provide fallback rates that are more robust.
- **Frequency Diversity:** Burst errors are avoided through coding, interleaving and scrambling in sub-carriers, thus protecting from deep-fading situations.
- **Multi-User scheme:** Each sub-carrier can carry data from different user without any modification. This scheme is referred to as OFDMA.
- **Robust against narrowband interference:** OFDM is relatively robust against narrowband interference, since such interference affects only a fraction of the subcarriers.

2.5.3 Applications of OFDM System

OFDM is equally famous in wired and wireless state of the art technologies to support high data rate communications and multi-user environment. Few examples are given below

- Broadband services like ADSL and VDSL
- Digital Audio and Video Broadcast Systems (DAB and HDTV)
- Broadband Wireless Access Systems and WLANs
    - IEEE 802.16
    - Hyperlan/2
    - IEEE 802.11 a/g
    - ITUT G.hn
- Multiple Access OFDMA in 3.5G and 4G mobile networks
- Personal Area Network (PAN) and Ultra Wide Band (UWB)
    - IEEE 802.15.3a
- Mobile Broadband Wireless Access (MBWA)
    - EEE 802.20
- Power line communication (PLC).

2.5.4 Peak to Average Power Ratio (PAPR) in OFDM

In OFDM system, all the multiple sub-carriers are summed up. There are chances that for certain OFDM symbols, constructive interference may take the power value very high. To measure this power increase, the term Peak to Average Power Ratio (PAPR) is used. It is the ratio of highest power value in the signal to average or expected value of that OFDM symbol. Mathematically:

$$PAPR = \frac{(max_i |x_k|^2)}{E[|x_k|^2]}$$

Here, xk is the signal at the transmitter, max shows that peak value is collected. E [ ] shows average expected value of symbol xk. This ratio must be kept as low as possible.

2.5.4.1 Challenges of PAPR

The issues raised through this phenomenon are as follows.

2.5.4.1.1 Operating Limits of Power Amplifiers

It is compulsory for power amplifiers to operate inside the linear region to protect inter-modulation of sub-carriers, radiation out-of-band and chances of damage of amplifier due to over voltage. These peaks can drag the amplifiers out of linear regions and damaging behavior will be displayed.

2.5.4.1.2 Battery Requirement for mobile devices

OFDM technology is widely used in wireless and mobile devices. Battery of these mobile devices is very limited. If PAPR value is high, more energy will be consumed and batteries will finish more quickly. For long life of batteries and, consequently, longer system alive time, we must design systems with least PAPR.

2.5.4.1.3 Range of Multicarrier Transmission

By limiting the peak power value, we are also minimizing the allowed average OFDM symbol power relating to constant modulation techniques. This in-turn causes reduction in transmission range of multi-user / multi-carrier systems.

2.5.5 Reduction in PAPR

Since the beginning of PAPR concept, engineers and scientists are trying different methodologies to reduce the same and avoid above mentioned issues. Few reduction schemes are listed

- Amplitude Clipping and Filtering
- Partial Transmit Sequence (PTS)
- Selected Mapping Technique (SLM)
- Interleaving
- Tone reservation and tome induction techniques.
- Tone injection techniques.

- Coding Technique
- Pre-Coding Technique.

2.6 Summary

In this chapter a brief introduction of digital signal processing is discussed with brief review of time domain and frequency domain properties and computations. Theoretical, mathematical and diagrammatic description of FIR filter, its responses and implementation challenges are discussed. An insight to single carrier and multicarrier systems and principles is given. OFDM system applications, advantages and the issues are discussed.

Chapter 3

Literature Review

3.1 Distributed Arithmetic (DA) Architecture

The idea of Distributed Arithmetic (DA) architecture as a replacement for MAC operation was first introduced by A. Croisier, D.J. Esteban, M.E. Levilion and V. Rizo, in U.S. patent named "Digital Filter for PCM Encoded Signals" [1]. Stanley A. White [2] explained in detail this idea of using the Distributed Arithmetic (DA) architecture with reference to applications in DSP applications. Whole techniques pivots on the assumption that one operand must be constant.

Distributed Arithmetic (DA) is visualized as a computation involving bits entering the system in serial fashion. The aim then becomes to calculate the dot product of two vectors, one being a fix valued vector and other being the variable input to the system, in a single direct step. All this is made possible by pre-computing the partial output values as one of the operand is fixed and other one is just a series of bits having two possible values, "0" or "1". As an example, consider the following Sum of Products (SOP) calculation

$$y = \sum_{k=1}^{K} A_k \, x_k \tag{3.1}$$

Here

- y is the out put
- A_k are the fix valued coefficients
- x_k are the variable input data words

Now if the inputs x_k are the 2's complement binary numbers then it can be expressed as

$$x_k = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} \, 2^{-n} \tag{3.2}$$

Here

- x_k are the variable input data words

- b_{k0} is the sign bit
- b_{kn} are the bits having values 0 or 1
- $b_{k,N-1}$ is the Least Significant Bit (LSB)

By putting the value of x_k from equation 3.2 into equation 3.1 we get

$$y = \sum_{k=1}^{K} A_k \left[-b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n}\right] \quad (3.3)$$

This is the conventional way to express inner product. Opening the brackets and then re-arranging the terms in equation 3.3, we get

$$y = \sum_{n=1}^{N-1} \left[\sum_{k=1}^{K} A_k b_{kn}\right] 2^{-n} + \sum_{k=1}^{K} A_k(-b_{k0}) \quad (3.4)$$

This completes the mathematical form necessary for Distributed Arithmetic computation. Focusing on the term inside the bracket in Equation 3.4

$$\sum_{k=1}^{K} A_k b_{kn}$$

Here, b_{kn} can have only two possible values, 0 or 1. This leaves the above term with only 2^K possible outcomes. These known 2^K values can be pre-computed and saved in the ROM or LUT instead of calculating in every cycle. The input data is used as the address to one of the 2^K memory locations present in ROM or LUT. So, the result $\sum_{k=1}^{K} A_k b_{kn}$ is added into the accumulator. After precisely N cycles, the final result y is achieved. Considering the above example, the ROM has to be 2×2^K deep because we have to cater for the negative sign bit as well as positive sign bit, shown as Ts. Considering that $K = 4$ we must have $2 \times 2^K = 32$ values in the ROM.

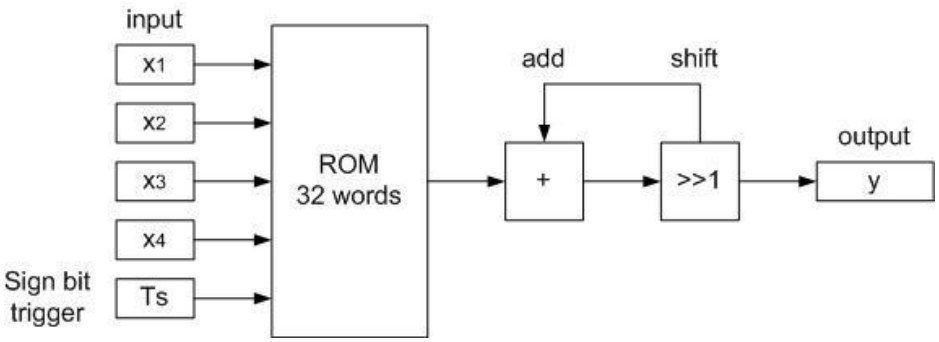


Figure 3.1: Basic idea for bit-wise DA Architecture

The contents inside the ROM are shown in Table 3.1

with +ive sign bits					
SR.#	Ts	b1n	b2n	b3n	b4n
0	0	0	0	0	0
1	0	0	0	0	1
2	0	0	0	1	0
3	0	0	0	1	1
4	0	0	1	0	0
5	0	0	1	0	1
6	0	0	1	1	0
7	0	0	1	1	1
8	0	1	0	0	0
9	0	1	0	0	1
10	0	1	0	1	0
11	0	1	0	1	1
12	0	1	1	0	0
13	0	1	1	0	1
14	0	1	1	1	0
15	0	1	1	1	1

with -ive sign bits					
SR.#	Ts	b1n	b2n	b3n	b4n
16	1	0	0	0	0
17	1	0	0	0	1
18	1	0	0	1	0
19	1	0	0	1	1
20	1	0	1	0	0
21	1	0	1	0	1
22	1	0	1	1	0
23	1	0	1	1	1
24	1	1	0	0	0
25	1	1	0	0	1
26	1	1	0	1	0
27	1	1	0	1	1
28	1	1	1	0	0
29	1	1	1	0	1
30	1	1	1	1	0
31	1	1	1	1	1

(Table 3.1 ROM Entries in Basic DA)

3.1.1 Improved DA architecture with less ROM entries

Stanley A. White [2] has also introduced a little modification. Instead of using the Ts as an input with like the data bits, replace the adder by add / subtract operation and use Ts as trigger to either add or subtract in case of normal bits and sign bits. This reduces the ROM Entries to 2^K only.

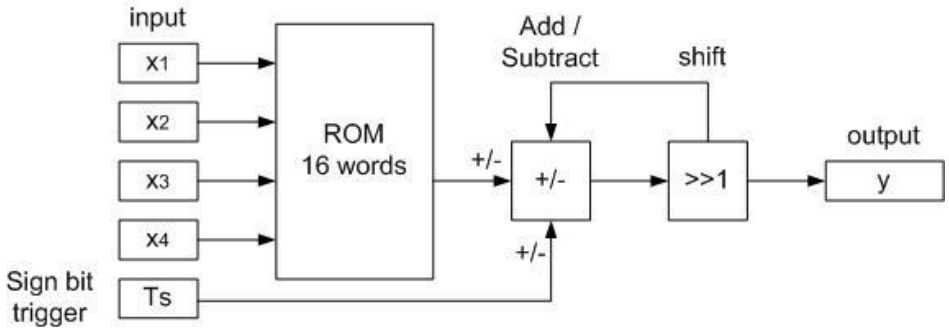


Figure 3.2: DA Design with less ROM

The entries are now half as much as were in initial design.

SR.#	b1n	b2n	b3n	b4n
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

(Table 3.2 ROM Entries in basic DA)

3.2 FIR Filter Design using DA Architecture

Extending the same idea of Inner Product calculation, the implementation of FIR Filter through Multiplier less architecture of Distributed Arithmetic (DA) is discussed by W. Sen, T. Bin, Z. Jun [5]. The equation of FIR Filter is

$$y[n] = \sum_{l=0}^{L-1} h[l]x[n-l] \qquad (3.5)$$

Re-arranging in the same way, diagram explains the working of DA architecture for FIR filter.

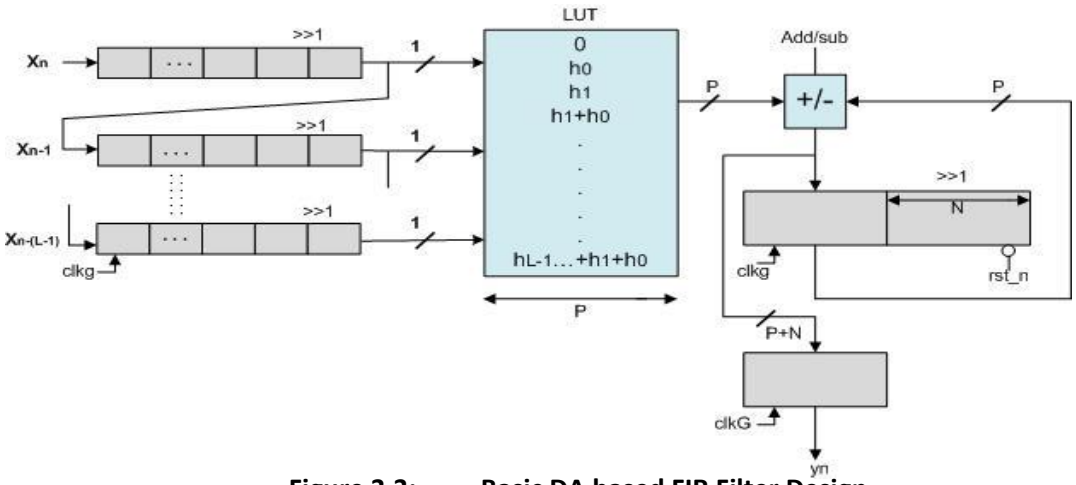


Figure 3.3: Basic DA based FIR Filter Design

P. Longa and Ali Miri [4] introduce Distributed Arithmetic (DA) architecture. Simple modification is introduced in the accumulator design and found improved area consumption on the hardware. The problem of increasing size of LUT remained unaddressed.

3.2.1 Direct Input DA based design

S. Hwang, G. Han, S. Kang, J. Kim [11] studied the technique with the perspective of reducing the power consumption and has devised a way to use the input bits without 2's complement. The designed architecture is shown in figure

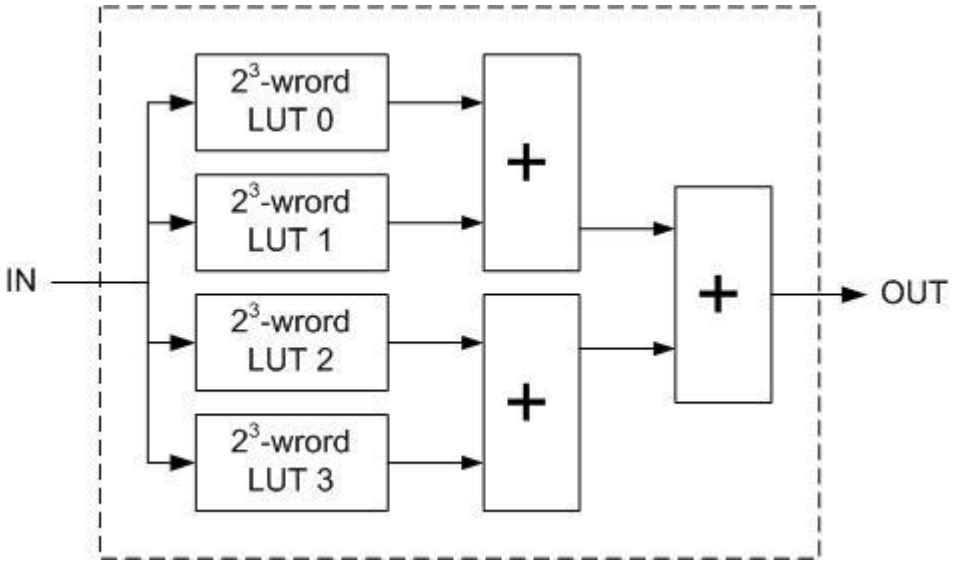


Figure 3.4: Direct Input DA Implementation

The results provided in this publication are given in table 2.3. Here also the main issue is that the increasing size of LUT with the increase in filter order is not discussed. Also the number of adders increase and extra logic will be required to cater for the negative signed values.

Parameters		Conventional Filter	Proposed Filter
Gate count		10,647	10,344
Power (mW)	Random Input (1.2288 MHz)	12.1659	8.6962
	Speech Signal (44.1 kHz)	0.3877	0.2883

(Table 3.3)

3.2.2 LUT less DA based design

H. Yoo and D. V. Anderson [3] has described DA based FIR filter implementation without any LUTs. The LUTs are replaced by the MUX and simple selection is required for the generation of partial product. The architecture for 4-tap FIR filter implementation is given in figure. This implementation can work on high frequencies. Number of adders required will be large to complete all the combinations causing extra delays.

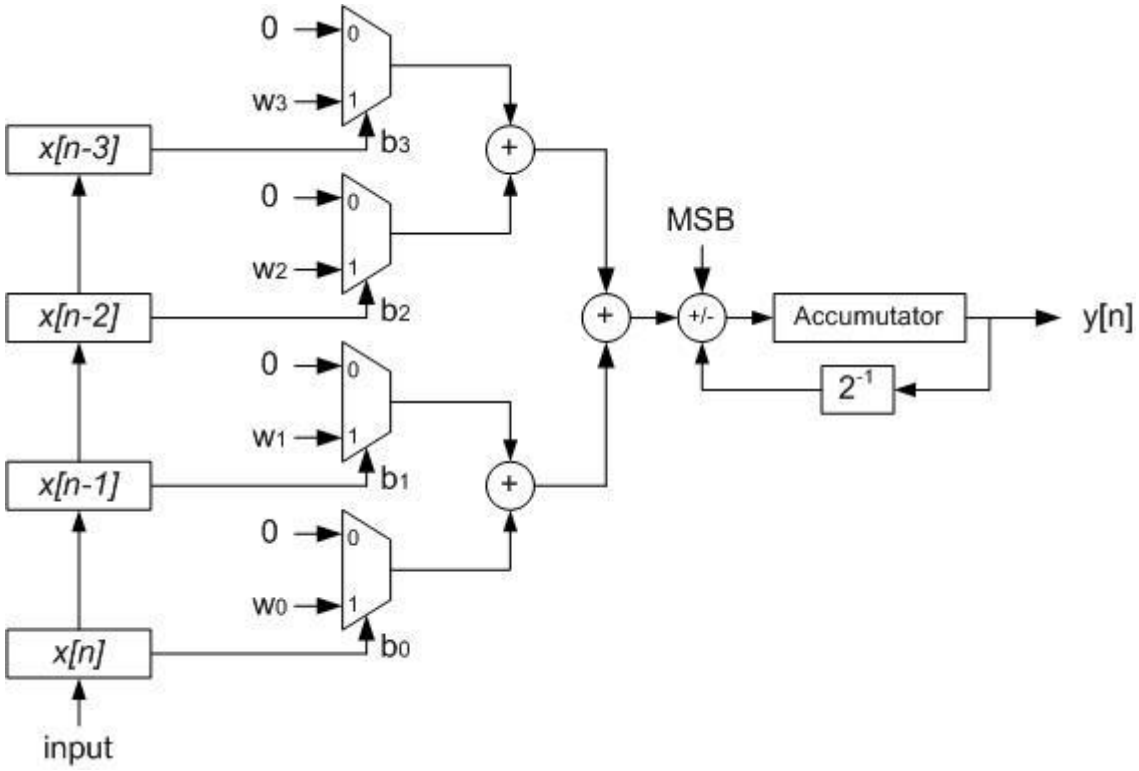


Figure 3.5: LUT-Less DA based Architecture for FIR Filter

3.2.3 M-Parallel Sub-Filter based design

W. Sen, T. Bin, Z. Jun [5] has introduced another efficient DA implementation modification to overcome the shortcomings. The size of LUT grows exponentially if the order or FIR filter is increased. This causes the design to get very heavy on the storage and increases the latency. The solution proposed is to break the LUT into equal parts and apply the same DA architecture. With the increase of order of FIR Filter, the partial products also increase in number this means the addition at the accumulator becomes the bottleneck.

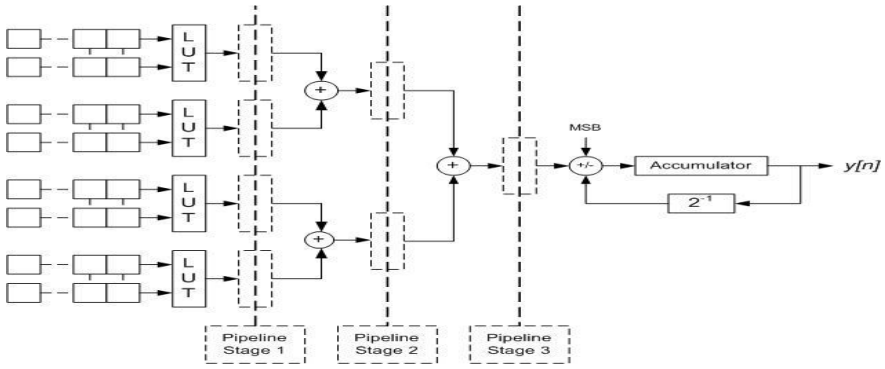


Figure 3.6: DA architectures for a K-tap (k=L X 4) FIR filter

Designs\parameters	Slice	4-input LUT	Max Frequency
Original DA Design	2,960	4,014	223 MHz
New DA Design	2,051	3,390	203 MHz

(Table 3.4)

The results provided show that the number of LUT is reduced as compared with the single LUT design which is the main focus. Multiple partial products need to be added causing minor decrease in operating frequency.

3.3 Equalizer design using FIR filter

J. Park, B. Richards, B. Nikolić [6] discussed a complete model for Single Carrier (SC) digital communication system. The transmitter model is given in figure 3.7

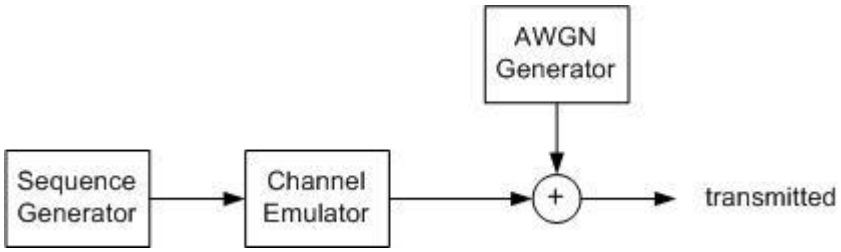


Figure 3.7: Transmitter Block Diagram

The equalization design at the receiver is described in figure 3.8

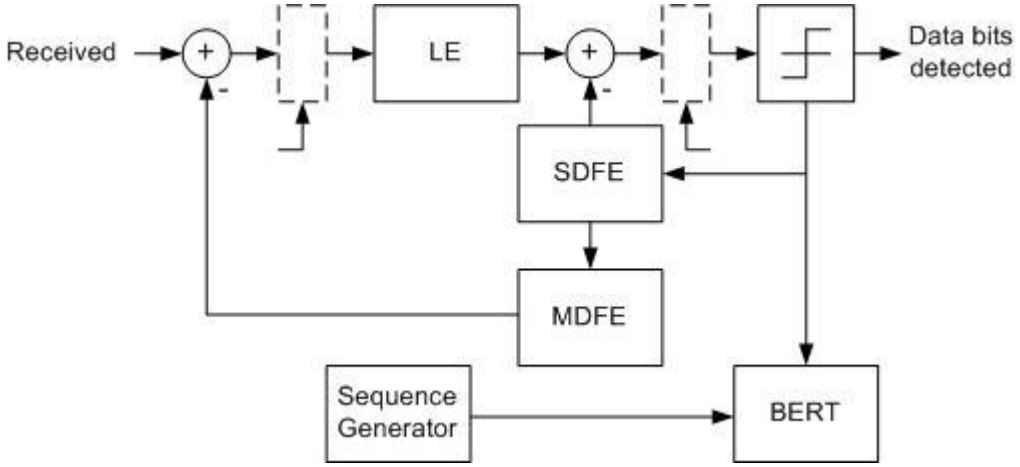


Figure 3.8: Equalizer and Channel Estimator Block Diagram

The most computational complex part of whole design is this equalizer part at the receiver that is the hybrid solution involving combination of three types of Equalizers

- Main Decision Feedback Equalizer (MDFE)
- Sub Decision Feedback Equalizer (SDFE)
- Linear Equalizer (LE)

Distributed Arithmetic (DA) architecture based FIR filters implement these equalizers. This avoids high computational complexity, increases the speed and makes it very easy to implement. Further to increase the throughput of the equalizers, 4-way parallel design is presented. The detailed architecture of equalizers discussed above is shown in figure 3.9

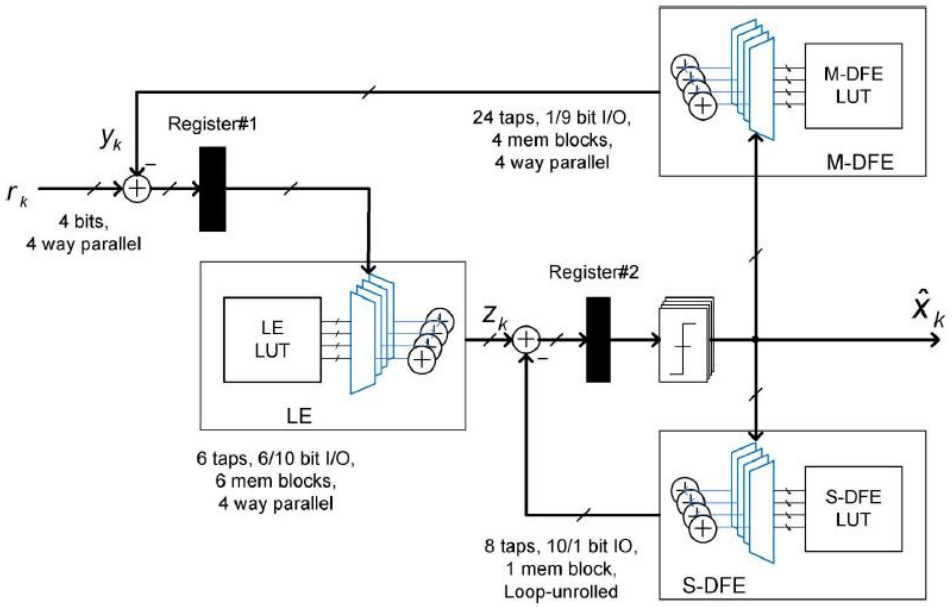


Figure 3.9: Equalizer Architecture with DA [6]

3.4 Precoding in OFDM system

Peak to Average Power Ratio (PAPR) is one of the major and most critical issues in the Orthogonal Frequency Division Multiplexing (OFDM) multicarrier digital communication system. The most efficient solution to this problem without compromising the precision of the data is addressed by Slimane Ben Slimane [7] through technique called Precoding.

A pre-defined matrix, designed through e-g- Root Raised Cosine Filter (RRC), is multiplied with every symbol before OFDM modulation (IDFT) at transmitter. This spreads the power to all the symbols hence reducing the Peak value to a considerably low value. At receiving end, same matrix is multiplied with the data after OFDM de-modulation (DFT).

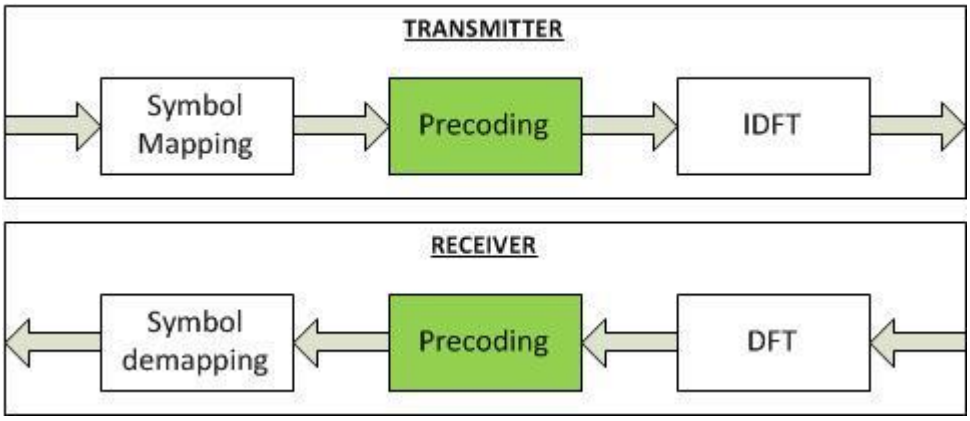


Figure 3.10: Precoding in OFDM System

For a system with N inputs, Np will be the overhead used and total size of precoded data will be L = N + Np. The equation (3.6) generates the first column constant vector based on RRC. This vector is multiplied to generate complete matrix equation (3.7).

$$p_{i,0} = \begin{cases} \frac{(-1)^i}{\sqrt{N}} \sin\left(\frac{\pi i}{2N_p}\right) & , \quad 0 \le i < N_p \\ \frac{(-1)^i}{\sqrt{N}} & , \quad N_p < i \le N \\ \frac{(-1)^i}{\sqrt{N}} \cos\left(\frac{\pi(i-N)}{2N_p}\right) & , \quad N \le i \le L-1 \end{cases} \tag{3.6}$$

$$p_{i,m} = p_{i,0} e^{-j2\pi \frac{im}{N}} \tag{3.7}$$

As an example, consider

- Data subcarriers in OFDM symbol = 32
- Pilot carriers for synchronization and channel estimation = 16
- Total subcarriers = N = 32 + 16 = 48
- Over-head used for precoding technique = Np = 16
- Total size of OFDM symbol L = N + Np = 64

This means that

- size of multiplicand matrix = 48 x 1 (column matrix)
- size of multiplier matrix = 64 x 48 (2D matrix)
- Number of multiplication operations = 48 x 64 = 3072
- Number of addition operations = 64 x 1 = 64

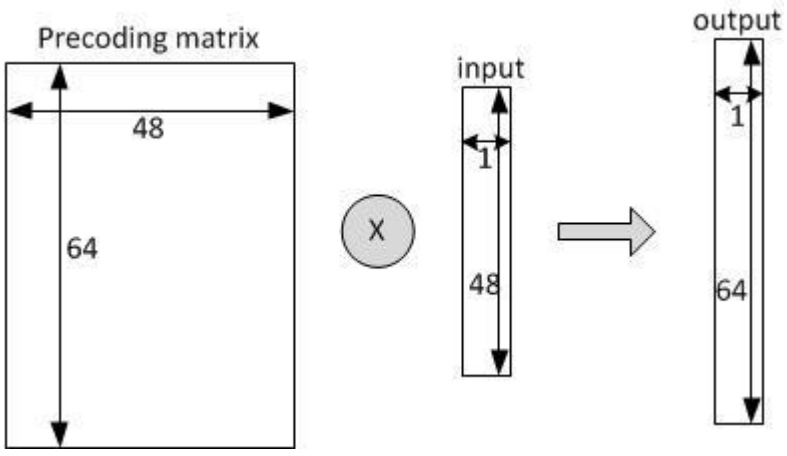


Figure 3.11: Data mapping on Precoded Matrix

This gives an idea of the complexity of the multiplication in the precoding technique. Now the only challenge remaining is the implementation of this multiplication as this has to be done both on the transmitting as well as receiving end. This clearly causes delay as well as exhausts resources of hardware Platform. The issue gets even worse for higher order OFDM system and must be addressed.

3.5 Summary

In this chapter the original theme of Distributed Arithmetic (DA) architecture is discussed in detail followed by various modifications from different authors to overcome shortcomings in previous designs. Supporting examples as well as architectural design diagrams are also provided. The use of Distributed Arithmetic (DA) in single carrier and need to introduce same in multicarrier OFDM system is described.

Chapter 4

Implementations and Modifications

4.1 Distributed Arithmetic (DA) Architecture

A complete functional 18-Tap FIR Filter is designed and implemented through Distributed Arithmetic (DA) architecture. The design is basically built on the M- Parallel Sub-Filter (MPSF) based Distributed Arithmetic (DA) architecture [5], being the most efficient in resource utilization. The parameters used are as follows

- K = No. of FIR filter coefficients
- M = No. of sub-filters
- L = Length of each sub-filter

The main challenge is the decision of number of sub-filters "M" to be used. For this decision, there are two aspects for which we have to take care, if any of them increases high then our design fails.

1. **Total space required as LUT memory**

   The number of LUT entries increase exponentially be decreasing the number of sub-filters used in the design. This has to be chosen very carefully because the total no. of entries in LUT will be saved in the memory of platform used and can cause deficiency for other processes. Mathematically it is equal to "M x (2^L)". This number must be kept as low as possible to occupy least possible space on the FPGA platform.

2. **Order of accumulator for addition**

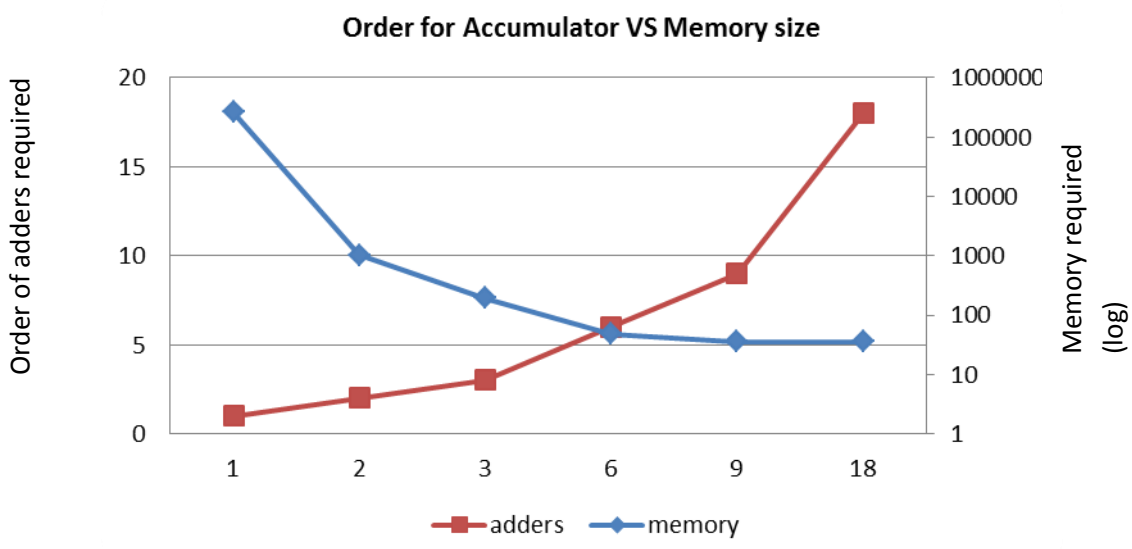
   Output from the LUTs is partial products which are added in an accumulator. Higher the number of sub-filters, higher will be the order of accumulator needed for addition. This will cause long delay and the critical path will also be long. So, in order to avoid this delay in

computations, least number of sub-filters must be deployed so that number of partial products is kept as low as possible.

These above two design factors are conflicting i.e. least memory usage requires highest “M” whereas, least partial products generation requires lowest “M”. Here a trade-off is to be made in the decision. All this discussion is summed up and analyzed in the Table 4.1. It shows all the possibilities for designing the FIR filter.

Moving from option#1 onwards as we keep on increasing the number of sub filters, i.e. we divide the LUT into more parts, the size of LUT memory required decreases at the cost of increase in the number of adders and order of accumulator required. In option#1 with all entries in a single ROM, we have 2^18 = 262,144 entries. Although 1 LUT will provide result but the amount of memory required is very huge.

This tradeoff can plotted as following. It is clear that with the increase of number of sub filters, the memory size requirement decreases and addition grows huge. These trends intersect at around 6 sub filters showing that maximum advantage is gained if we choose option#4.



The parameters chosen for implementation are

- K = 18 = No. of FIR Filter Coefficients
- M = 6 = No. of Sub-Filters
- L = 3 = Length of each Sub-Filter

(Table 4.1)

Analysis for No. of Sub-Filters "M"							
Option #	Order of FIR Filter	Number of Sub-Filters	Number of coefficients in each sub-Filter	Number of Entries in a single Sub-Filter	Total Number of Entries in all LUTs	Total Number of Partial Products Generated	Order of Accumulator Adder Required
	K	**M**	**L**	**2 ^ L**	**M x (2^L)**	**M**	**M + 3**
1	**18**	**1**	**18**	**262144**	**262144**	**1**	**4**
2	**18**	**2**	**9**	**512**	**1024**	**2**	**5**
3	**18**	**3**	**6**	**64**	**192**	**3**	**6**
4	**18**	**6**	**3**	**8**	**48**	**6**	**9**
5	**18**	**9**	**2**	**4**	**36**	**9**	**12**
6	**18**	**18**	**1**	**2**	**36**	**18**	**21**

This gives us the optimum solution and resolves the issue.

On the basis of the above discussion and choice made, the basic architecture to realize the 18-coefficient FIR filter through Distributed Arithmetic (DA) architecture is shown in figure (4.1).

It shows 6 LUTs, each catering independently for one of the M = 6 sub-filters. After that, the partial products are being added up and in case of the sign bit addition/ subtraction takes place.

All the designs are implemented and tested in Verilog and synthesized for Virtex-5 FPGA platform through Xilinx 12.1 synthesizer.

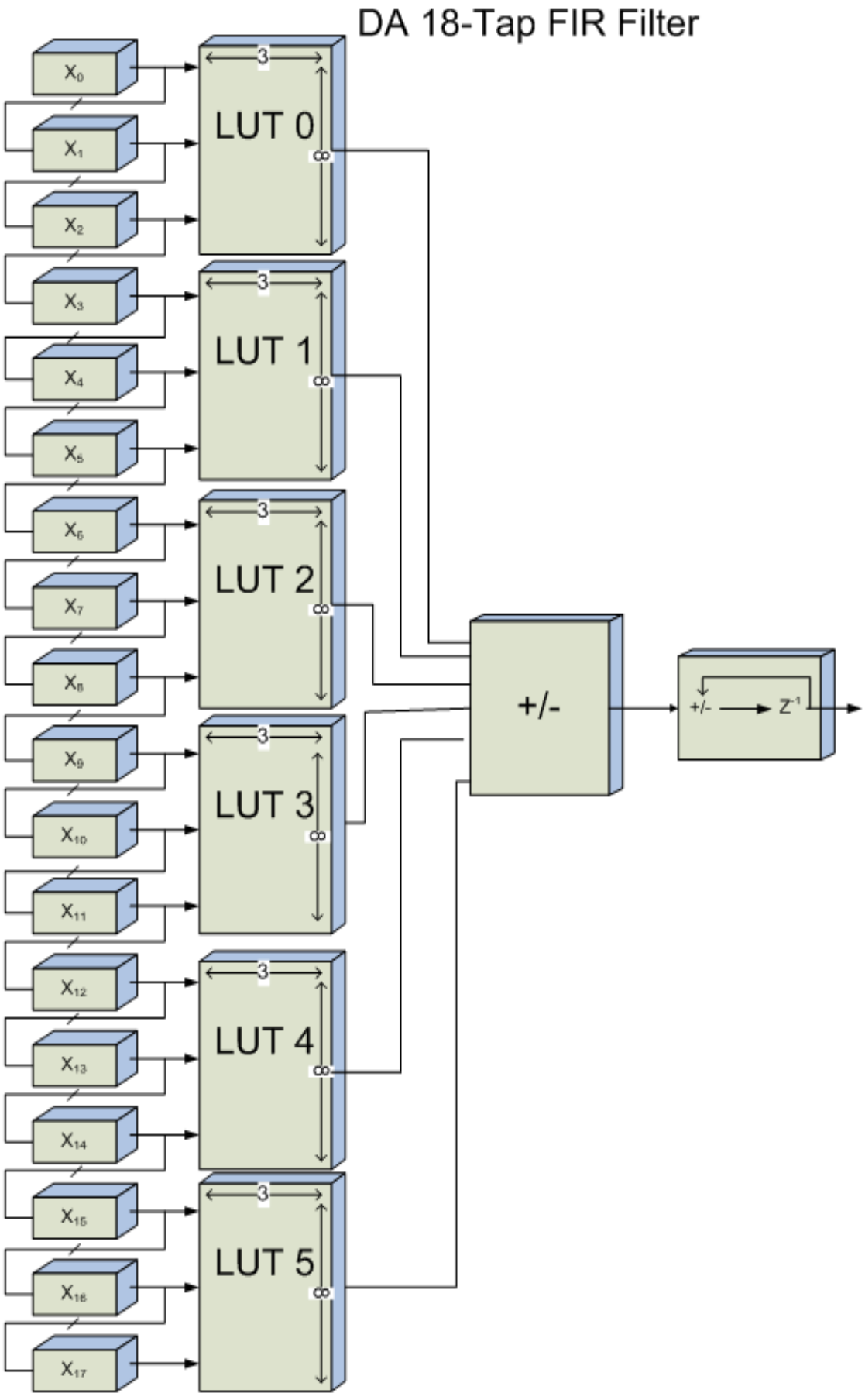


Figure 4.1: Realization of 18-Tap FIR Filter through DA

The bit wise flow of the implemented design for 18-Coefficient FIR Filter is shown in figure 4.2.

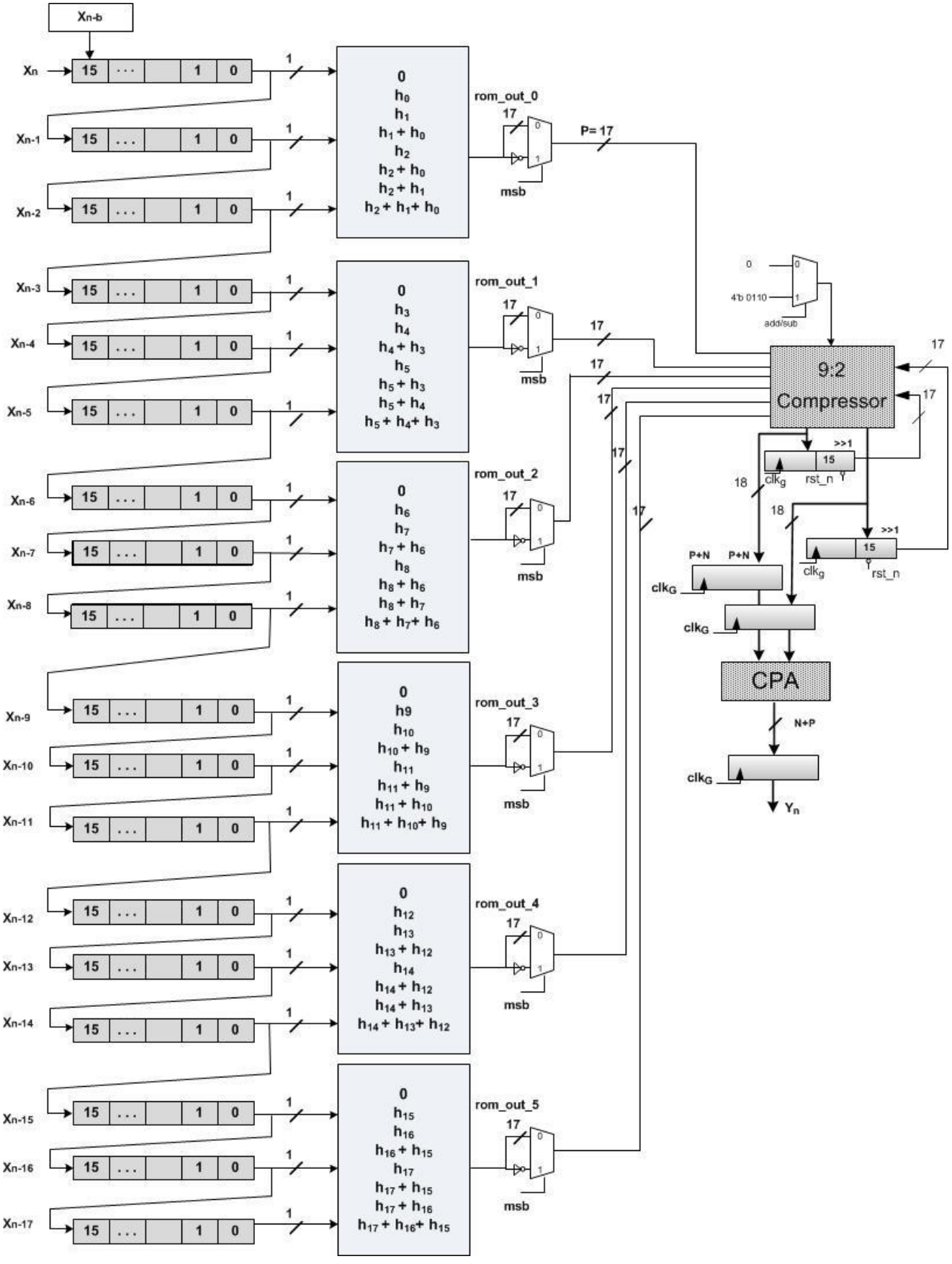


Figure 4.2: Bit-wise implementation of DA based 18-Coefficient FIR Filter with Daisy Chain Input String

4.2 Modified Implementation of DA based FIR filter

In the above described design of the DA based FIR filter, the number of partial products for addition is still relatively high, 9, which takes multiple iterations for adder to generate the final output. For efficient and fastest addition, following techniques can be used

- Wallace tree : this requires minimum 5 cycles to compute the final result
- Dada tree : this requires minimum 4 cycles to compute the result

To further reduce the number of cycles required for addition, a little modification is implemented shown in figure 4.3. The partial product that is used to cater for the sign bit i.e. to decide addition or subtraction can be skipped by using an AND gate on the counter and XOR gate on output of every LUT. In case of sign bit, the AND gate triggers the signal and accordingly XOR gate calculates the required partial product depending on the sign. Pipeline register after this XOR operation reduces the delay in the critical path at the cost of latency. Through this model, 8 partial products need to be added and thus following results can be obtained

- Wallace tree : this requires minimum 4 cycles to compute the final result
- Dada tree : this requires minimum 4 cycles to compute the final result

All the implementations are successfully synthesized and tested on Virtex-5 FPGA platform through Xilinx 12.1 synthesizer. Figure 4.4 to 4.8 are showing all this function stage wise by highlighting and description. Figure 4.9 shows the actual waveform for the data and control signals after implementation, synthesis and simulation.

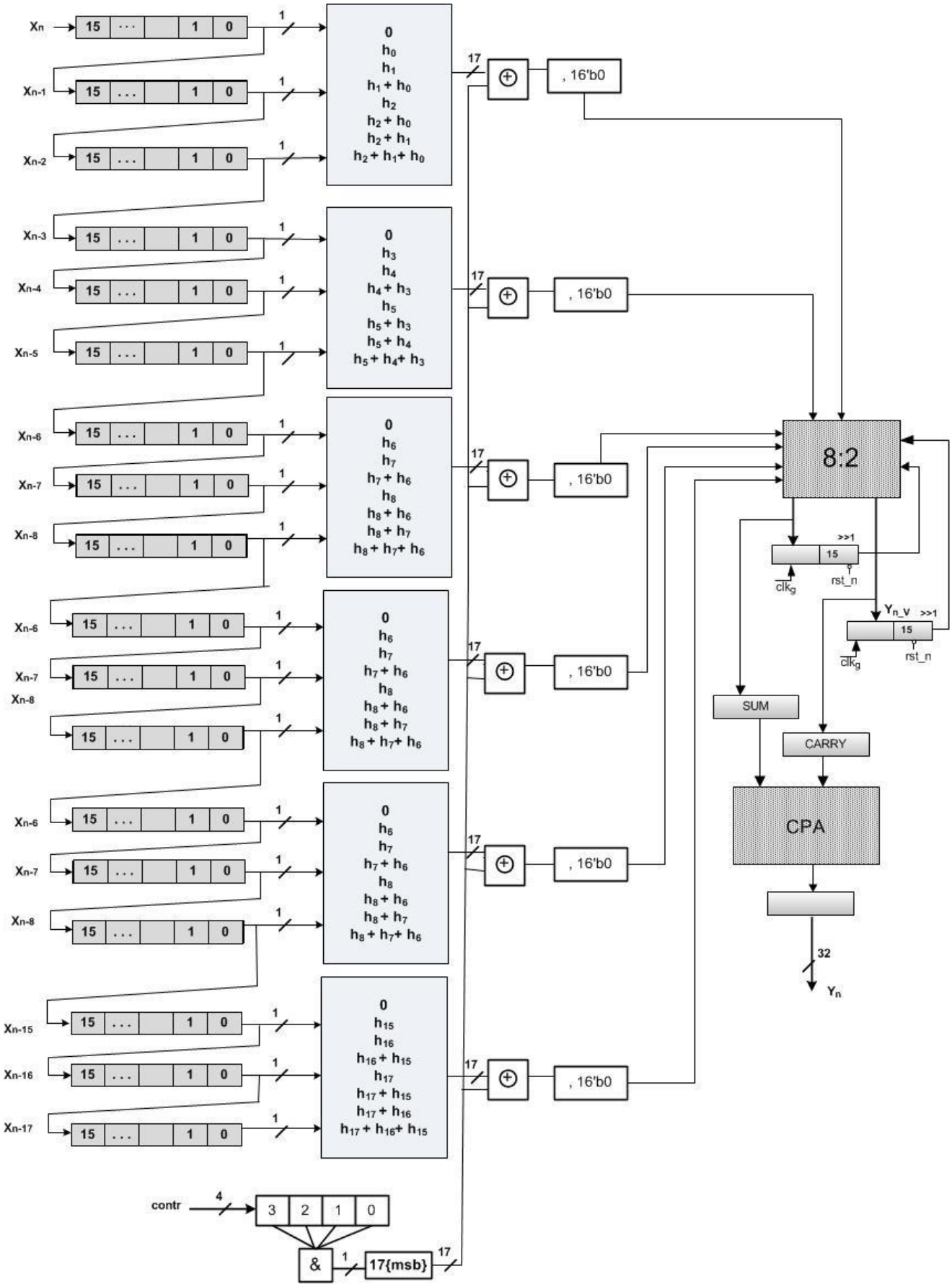


Figure 4.3: Bit-wise implementation of modified DA based 18-Coefficient FIR Filter with Daisy Chain Input String including XOR and AND

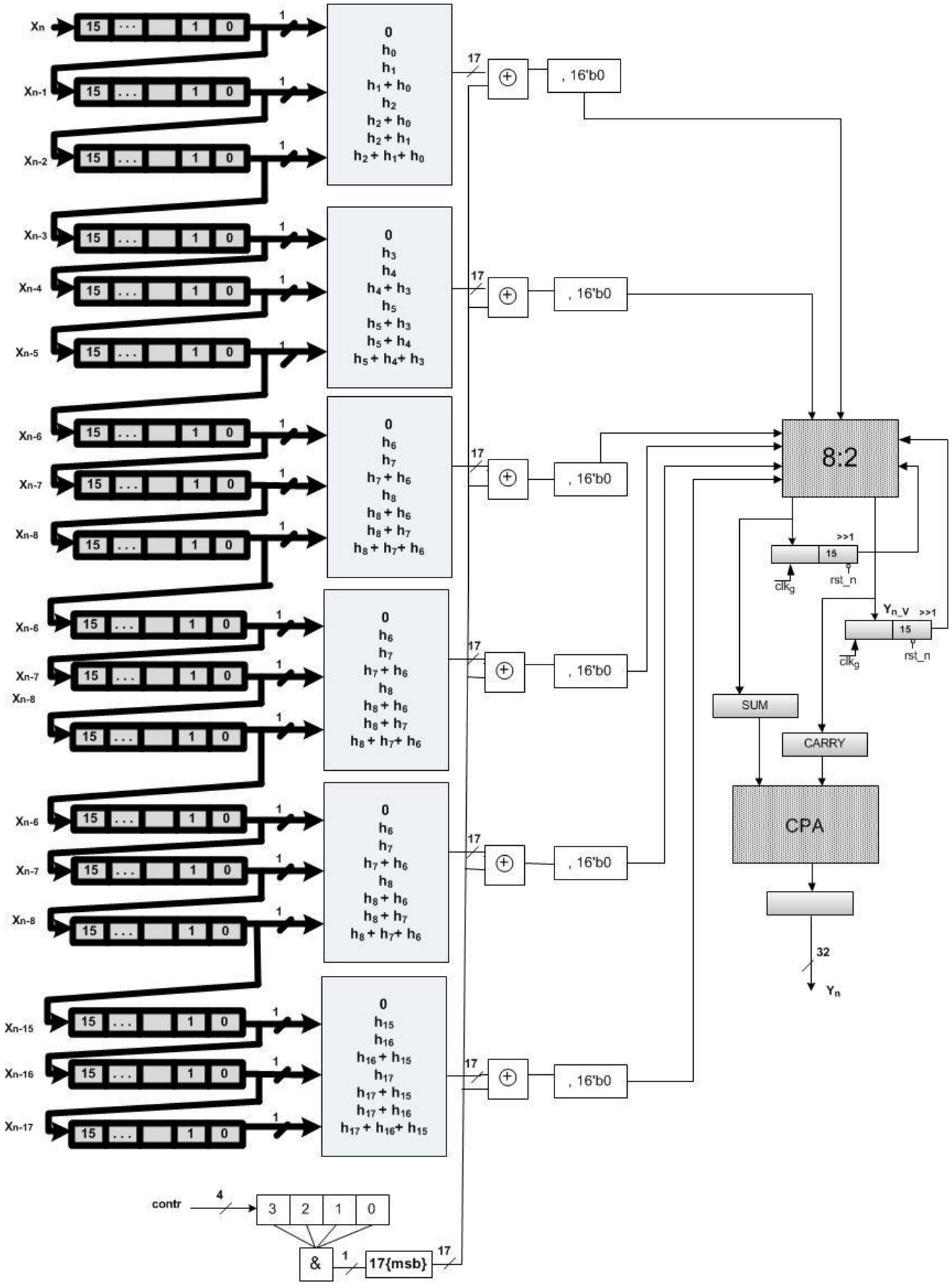


Figure 4.4: Stage-1: Bits entering the system start moving internally in a daisy chain method. LSB bits combine to form the address of corresponding LUT entry

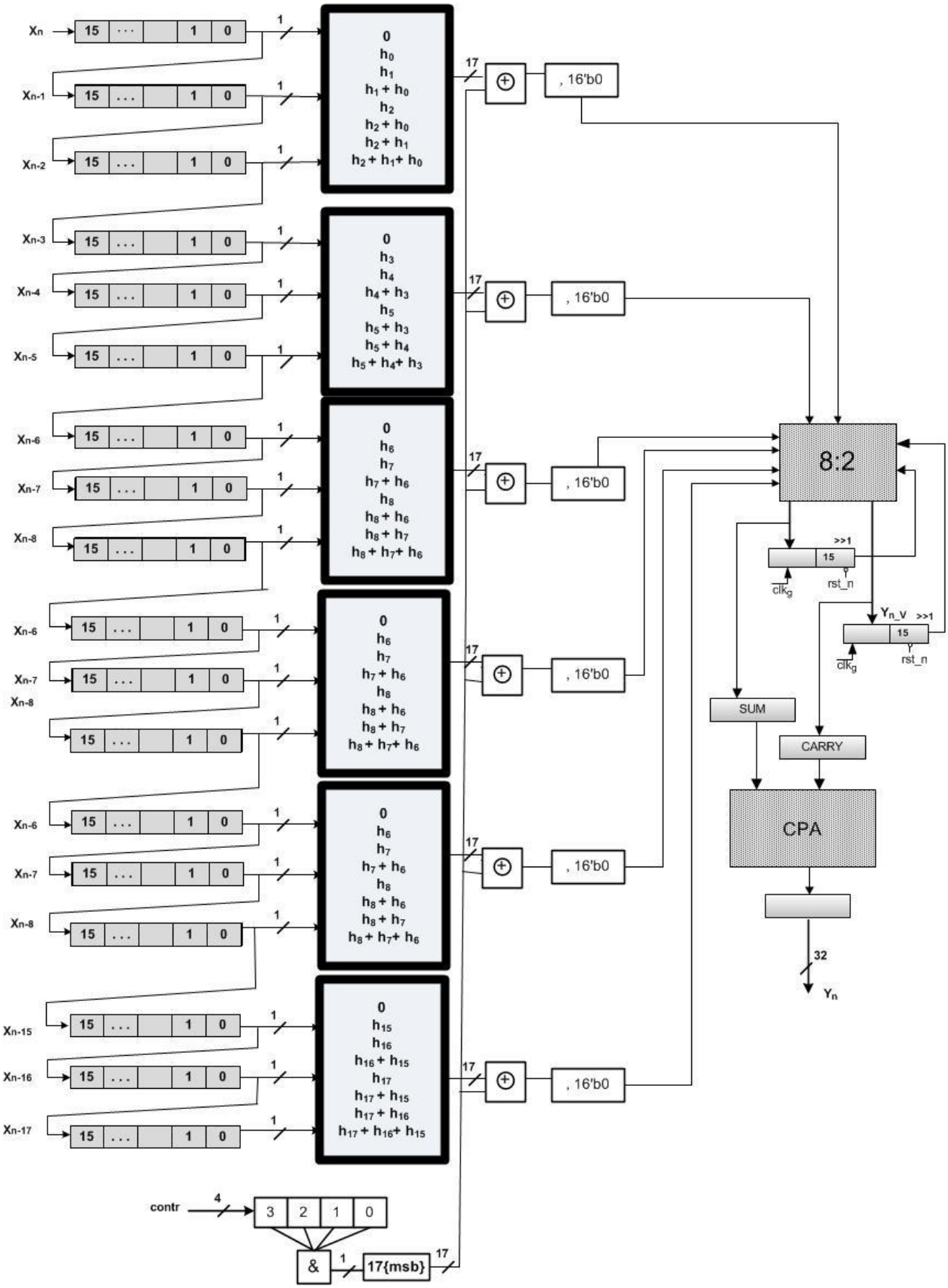


Figure 4.5: Stage-2: LUTs are containing the possible combinations of coefficients. Appropriate combination is selected according to bits in daisy chain and value is passed to next stage

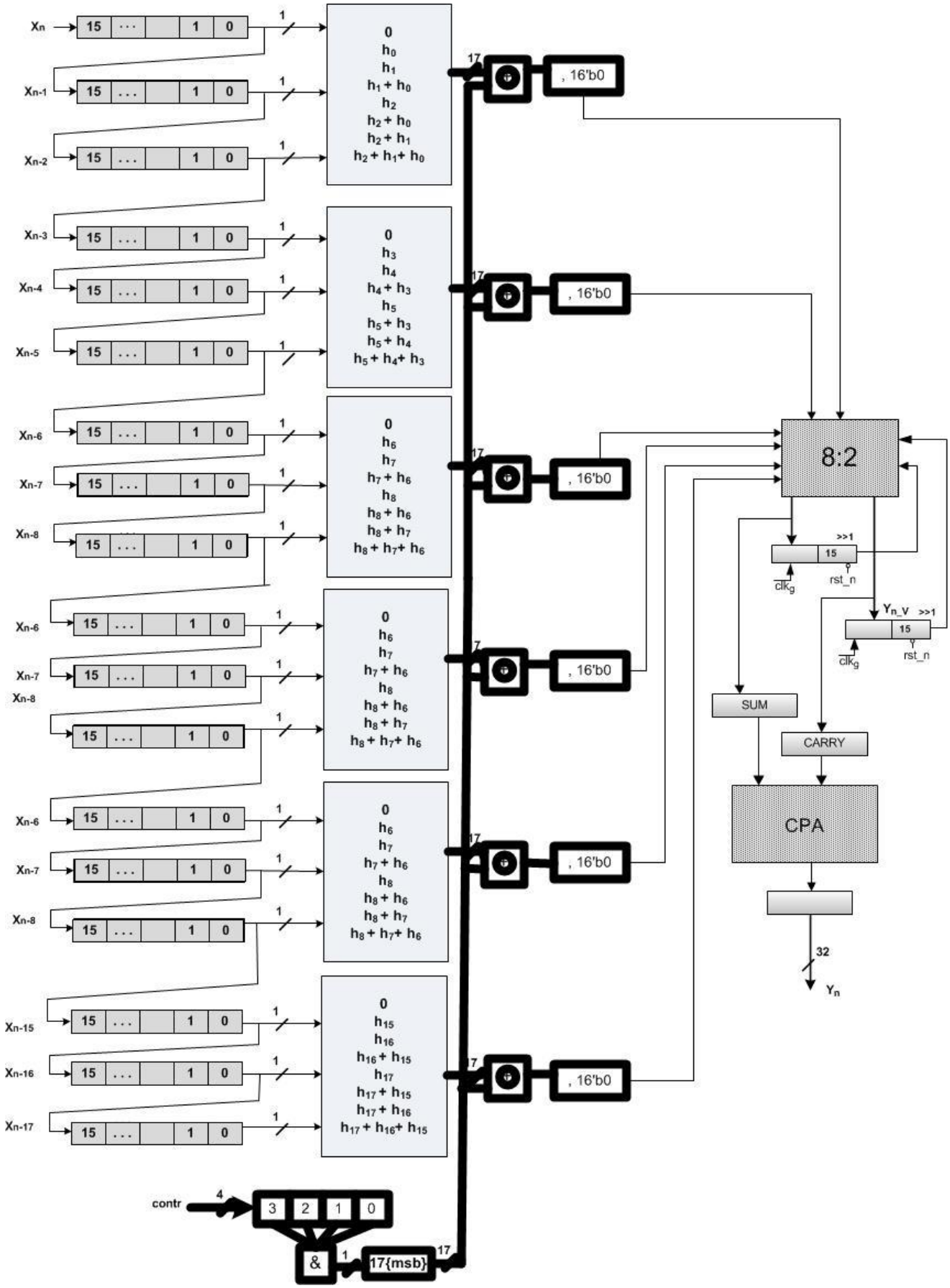


Figure 4.6: Stage-3: counter based on and gate is checking for the MSB. On hitting MSB, it triggers the logic for 2's complement and values are saved in pipeline registers

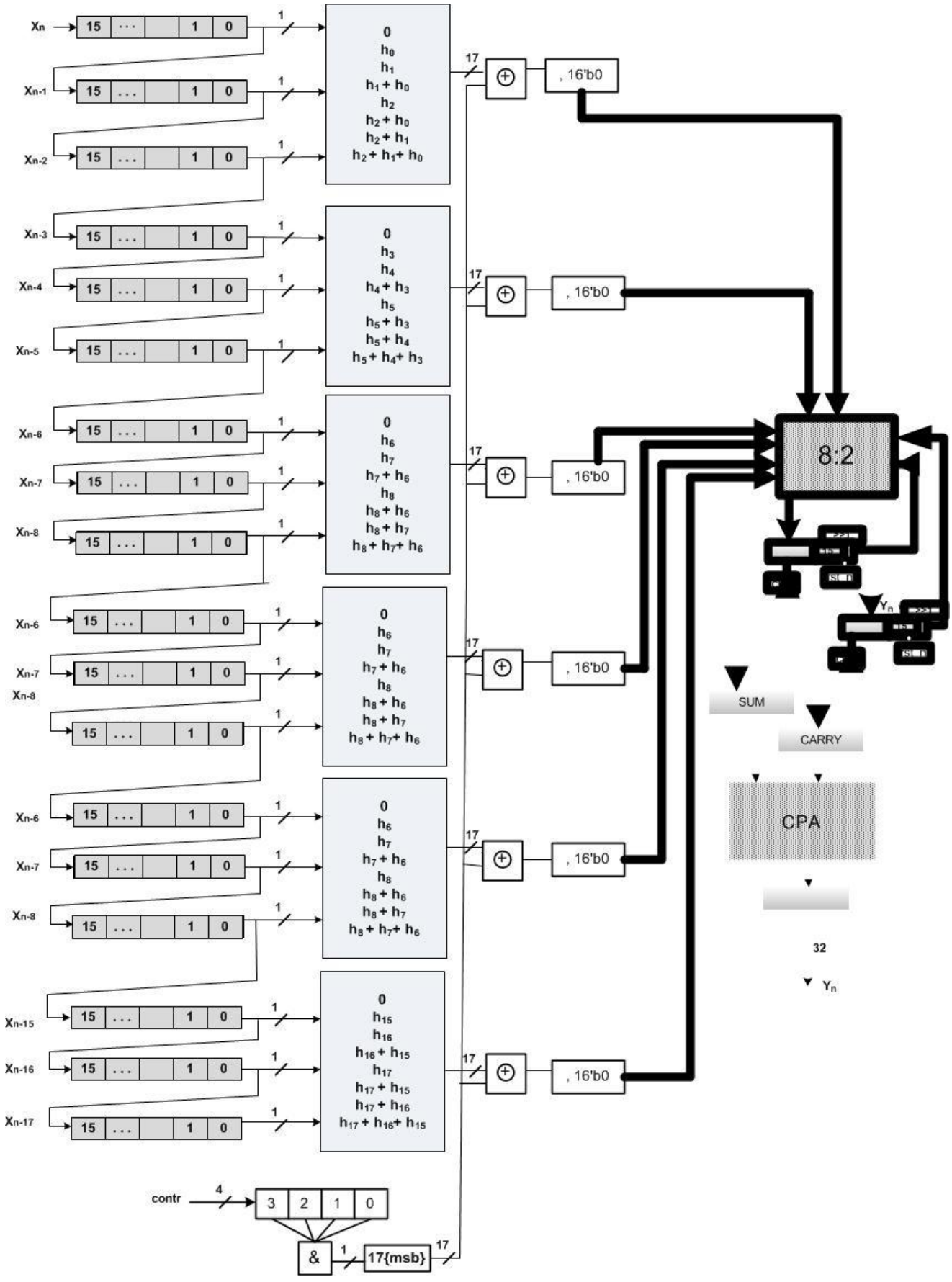


Figure 4.7: Stage-4: Pipeline registers forward the partial products to accumulator which gives output in form of sum and carry. These are resent to accumulator for next clock addition

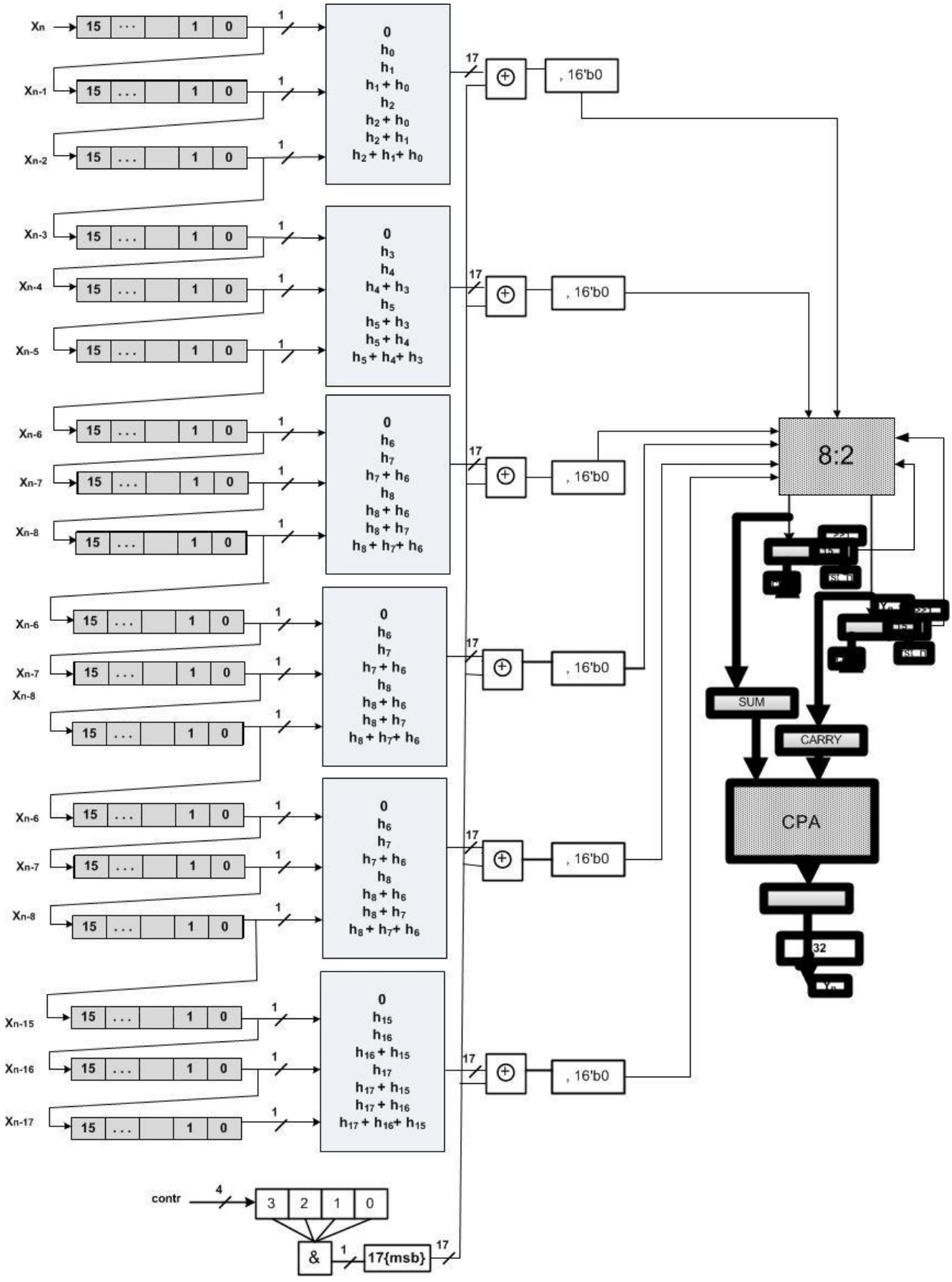


Figure 4.8: Stage-5: After the trigger of MSB, the last sum and carry are sent to carry propagate adder which provides final result

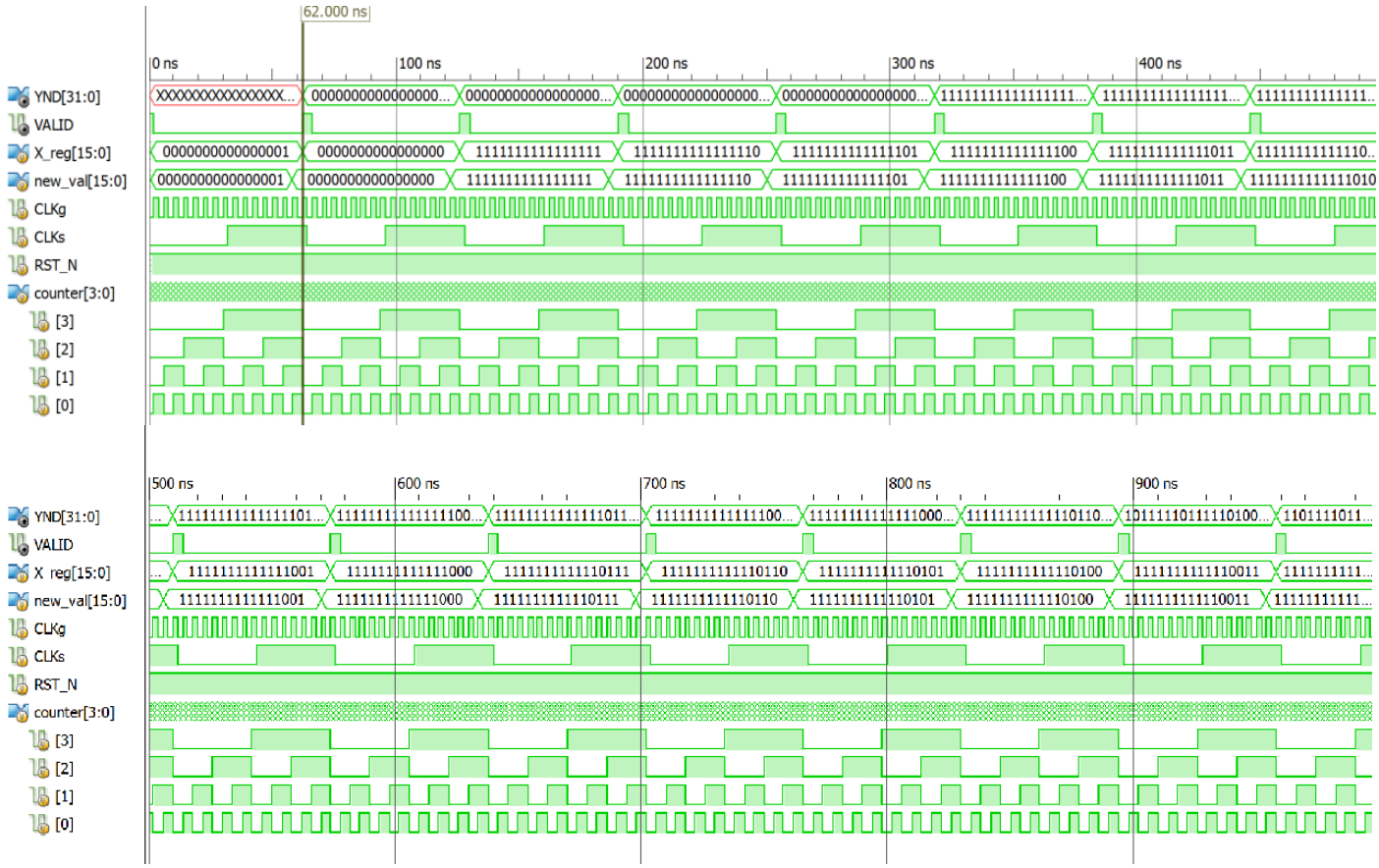


Figure 4.9: Synthesized waveform for all the involved inputs, outputs, counters and control signals in 18 tap FIR Filter implementation by DA architecture

Figure 4.9 shows the actual waveforms of all the data as well as control signals in DA based FIR Filter. Here Signal Representation is as follows

- YND = Output Signal
- Valid = Trigger Validity of Output after N(=16) inner clock cycles
- X_reg = Temp Input signal
- New_val = Input Signals
- CLKg = faster inner clock (CLKg = 16 x CLKs)
- CLKs = slower outer sample clock
- RST_N = reset for all registers
- Counter = to check MSB

On every rising edge of valid signal the output is ready. The sample clock CLKs is 16 times slower than the inner functional clock. The counter signal is correctly detecting the MSB signal for taking 2's complement for last partial product.

4.3 DA Based FIR implementing 4-way - Parallel Equalizer

The described FIR technique is used to implement the equalizers already described in section 3.3. The combination of three equalizers is used namely:

- Main Decision Feedback Equalizer (MDFE)
- Sub Decision Feedback Equalizer (SDFE)
- Linear Equalizer (LE)

These equalizers are designed by using FIR filter [6]. The DA technique discussed in section 4.2 is used to implement these equalizers. The equalizers are implemented in 4-way parallel fashion i.e. 4 hardware equalizers are functioning concurrently to increase the throughput up to 4 times.

4.3.1 Main Decision Feedback Equalizer (MDFE)

Mathematically, MDFE can be represented by equation 4.1

$$y_k = \sum_{m=1}^{24} h_{L+m} x_{k-m} \quad (4.1)$$

This direct form FIR can be implemented in parallel as in equation 4.2

$$y_{4q+p} = \sum_{m=1}^{24} h_{L+m} x_{4q+p-m} \quad (4.2)$$

The order of FIR filter is 24. This is further divided into 4 sub-filters each having 6 coefficients because the partial product minimization is more important. Each LUT contains 2^6 = 64 entries. The implementation block diagram is shown in Figure 4.4. The 4-way parallel implementation is successfully implemented and tested in Verilog and synthesized for Virtex-5 FPGA platform.

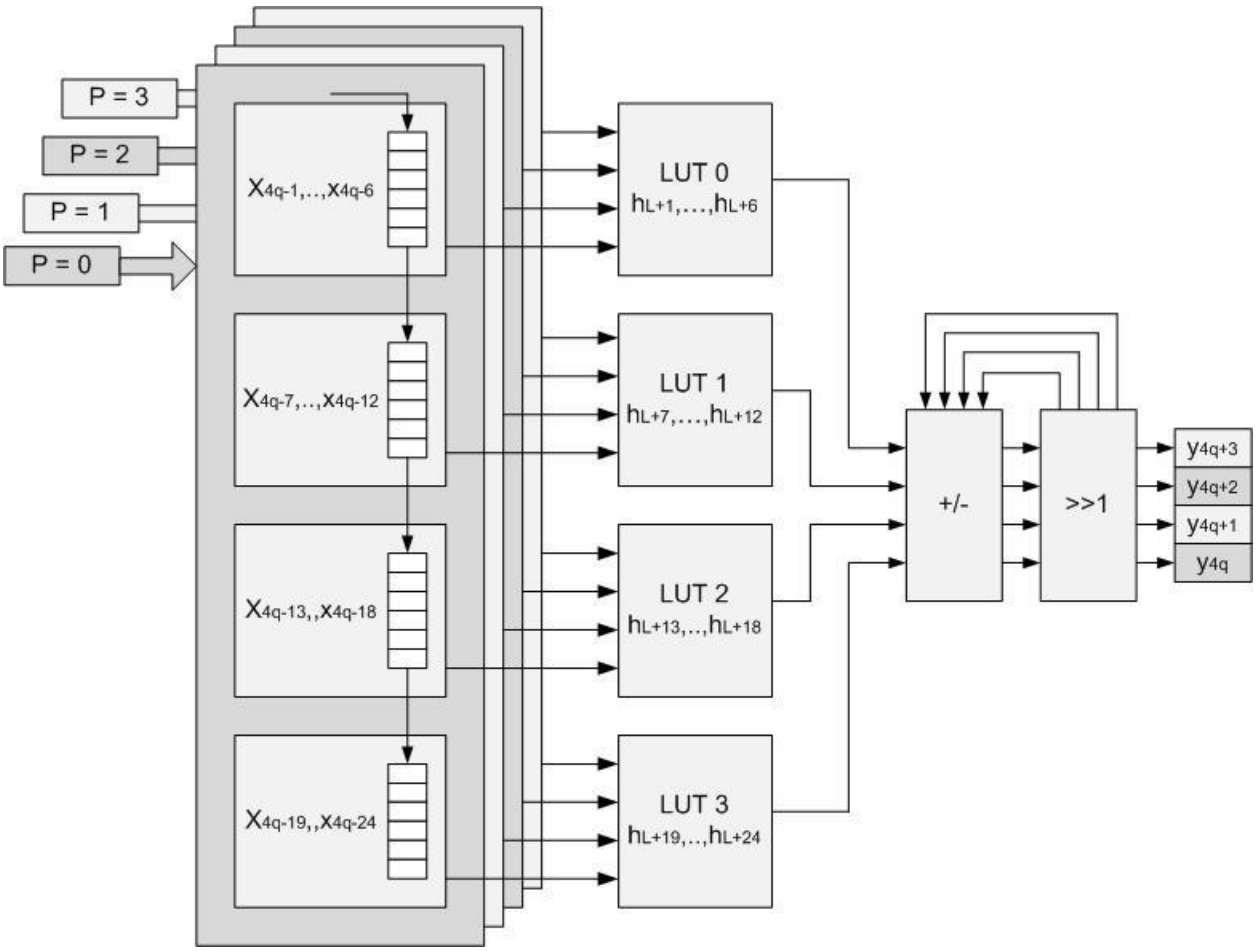


Figure 4.10: Main Decision Feedback Equalizer (MDFE) 4-way Parallel Implementation through Distributed Arithmetic (DA) based FIR Filter of order 24

4.3.2 Sub Decision Feedback Equalizer (SDFE)

SDFE implementation is a little different as it must support the feedback in a single cycle. Therefore the slicer and SDFE are combined and loop-unrolled to implement through DA architecture. The compromise is made at the cost of memory size as now we have to implement 8 coefficient SDFE by using only 1 LUT containing 2^8 = 256 entries. The implemented block diagram is shown in figure 4.5. This also caters for the 4 way parallel architecture design i.e. 4x inputs from Linear Equalizer (LE) and 4x outputs to the equalizers. The implementation is successfully implemented and tested in Verilog and synthesized for Virtex-5 FPGA platform through Xilinx 12.1 synthesizer.

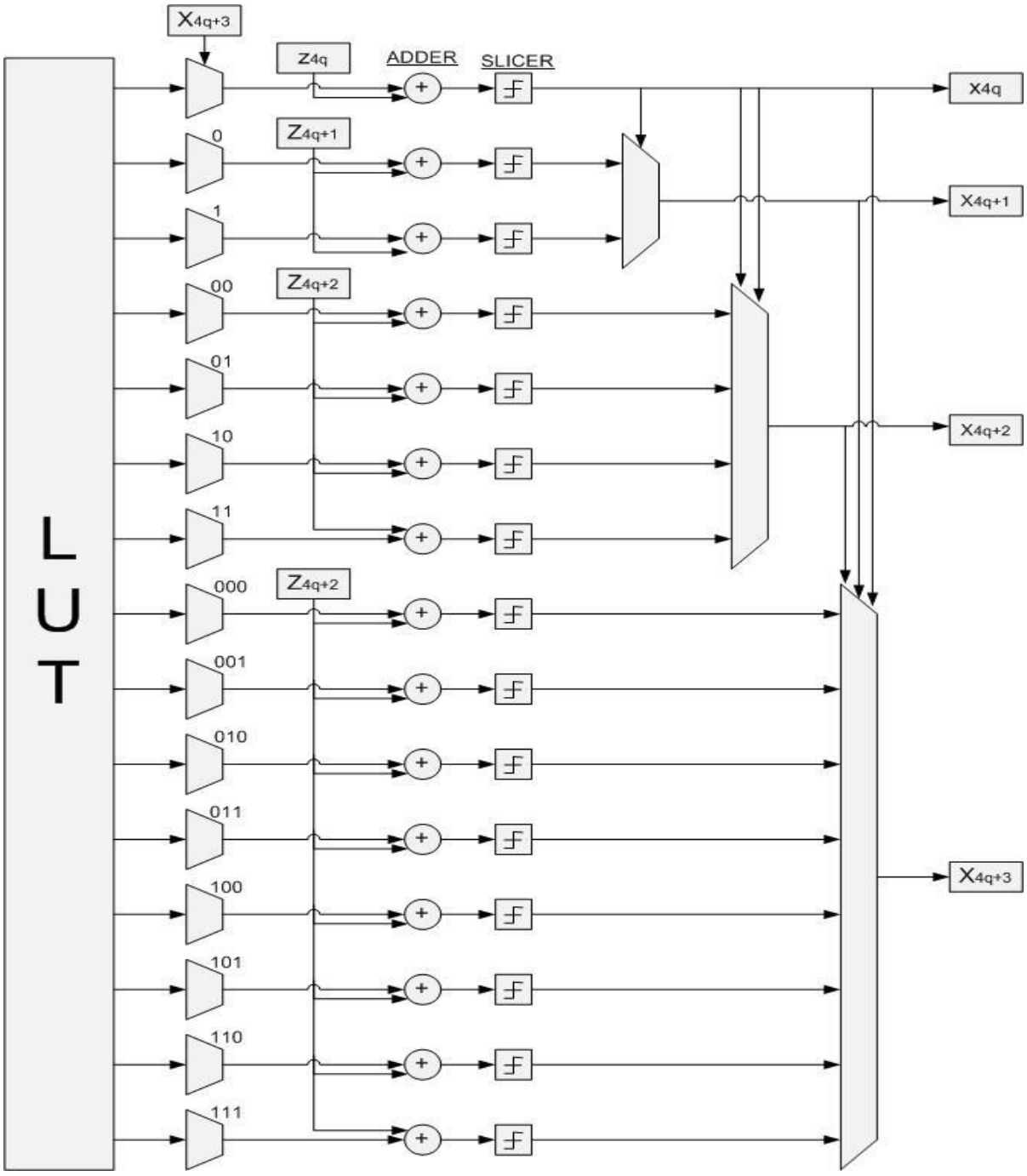


Figure 4.11: Sub Decision Feedback Equalizer (MDFE) to cater 4-way Parallel Implementation through Distributed Arithmetic (DA) based FIR Filter of order 8

4.3.3 Linear Equalizer (LE)

Linear Equalizer is also implemented by a 6 tap FIR filter through 1 LUT having 2^6 = 64 entries so that the partial products remain as low as possible, compromising the size of LUT. The 4-way parallel implementation is successfully synthesized and tested on Virtex-5 FPGA platform. The detailed block diagram is shown in figure 4.6.

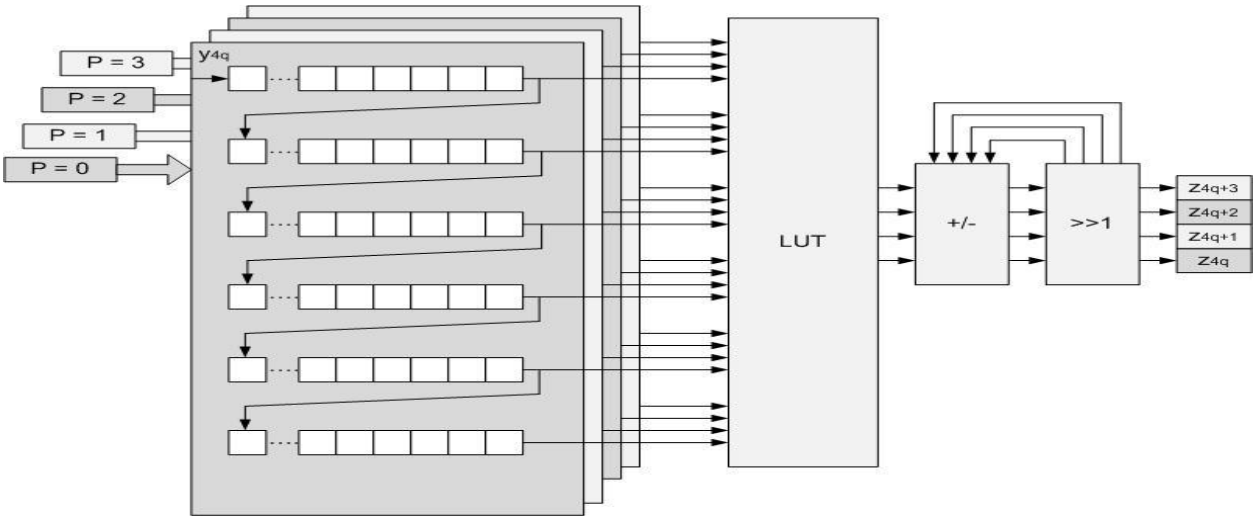


Figure 4.12: Linear Equalizer (LE) 4-way Parallel Implementation through Distributed Arithmetic (DA) based FIR Filter of order 6

4.4 Distributed Arithmetic (DA) for Precoding in OFDM System

Precoding involves multiplication of a fixed 2-dimentional matrix with variable column matrix. There is a set pattern for multiplication and each input needs to be sent to a particular concerned LUT only. The test implementation is based on following example for analysis purpose. All the values are supposed to be 3-bit wide.

- Total subcarriers = N = 3
- Over-head used for precoding technique = Np = 1
- Total size of OFDM symbol L = N + Np = 4

This means that

- Size of multiplicand matrix = 3 x 1 (column matrix)
- Size of multiplier matrix = 4 x 3 (2D matrix)
- No. of multiplications = 4 x 3 = 12
- No. of additions = 4 x 1

Following values are considered in test implementation

2-D Precoding matrix		columns #		
		0	1	2
rows #	0	1	6	3
	1	1	5	3
	2	4	7	2
	3	2	3	2

1-D Data In as OFDM Symbol		column #		
		0		
rows #	0	a[2]	a[1]	a[0]
	1	b[2]	b[1]	b[0]
	2	c[2]	c[1]	c[0]

There are 8 possible values in fixed precoding matrix as number of bits is fixed to 3. Therefore there will be 7 LUTs in total each for a possible value; LUT for “0” is not required. Each LUT will

further have 8 entries as there are in total 2^3 = 8 possible answers. The input is only routed to the concerned LUT as we already know one operand. The partial products are

Partial Products		column #			Output
		0			
rows #	0	a x 1	b x 6	c x 3	w
	1	a x 1	b x 5	c x 3	x
	2	a x 4	b x 7	c x 2	y
	3	a x 2	b x 3	c x 2	z

The detailed block diagram of implementation is shown in figure 3.7. This implementation does not involve any multiplier so we can easily use parallel architecture for more than one OFDM bursts at the same time, thus increasing the overall output of the system. The implemented design is successfully synthesized and tested on Virtex-5 FPGA platform through Xilinx 12.1 synthesizer.

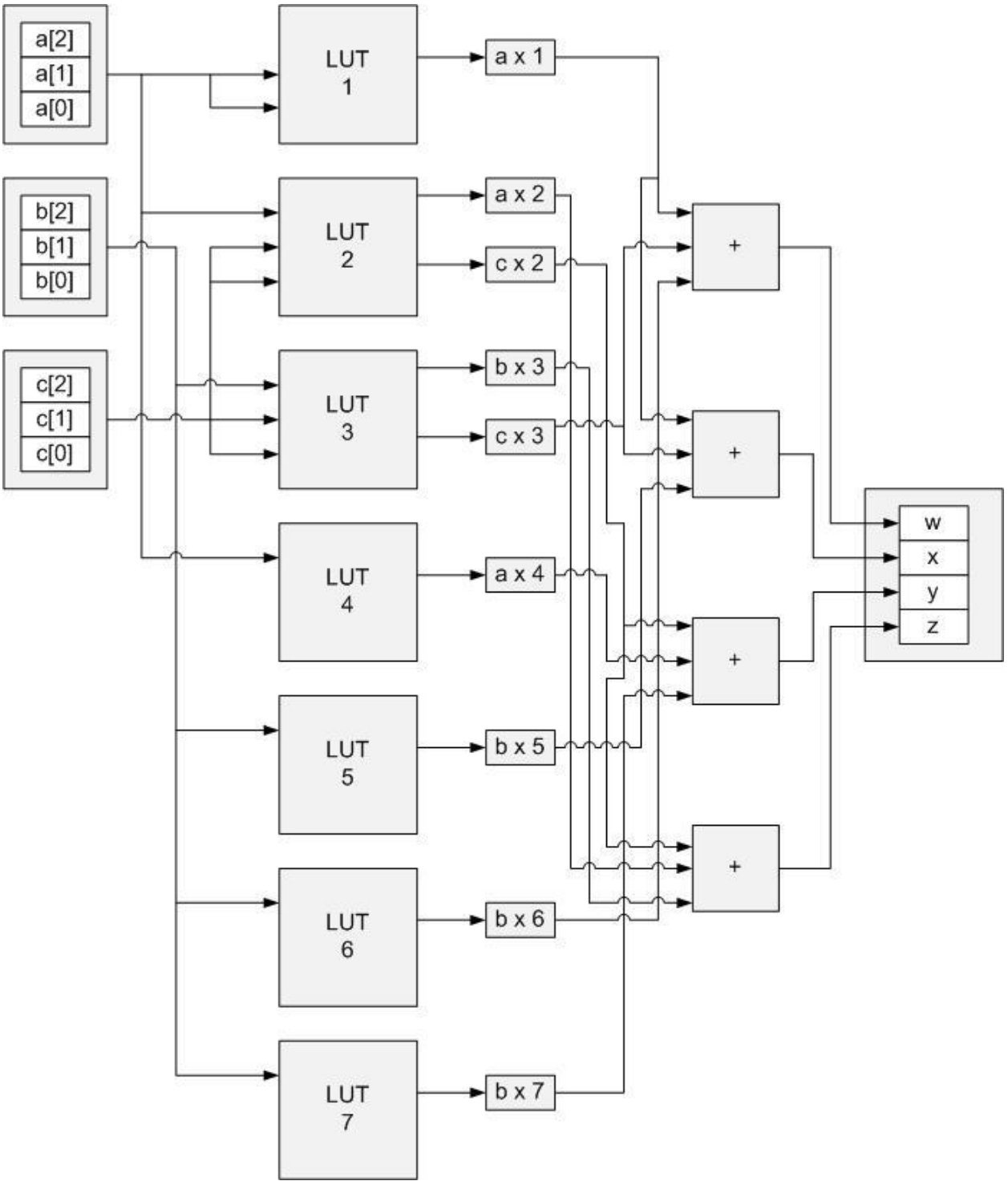


Figure 4.13: Precoding Multiplier-less Multiplication using Distributed Arithmetic (DA) Architecture

4.5 SUMMARY

In this chapter the design and implementation of FIR filter, Main Decision Feedback Equalizer (MDFE), Sub-Decision Feedback Equalizer (SDFE) and Precoding matrix multiplication are discussed in detail. The results follow in the next chapter.

Chapter 5

Resource Utilization Analysis

In the coming section, the designs and modifications discussed and implemented are studied with respect to the resources utilized. The comparison is made between the conventional designs and the modified implementations.

5.1 MACs and Multipliers

The implementation of modified DA based implementation focus on minimizing the resources especially the multipliers and MAC blocks as these are one of the most resource consuming operations. The DA based architecture completely eliminates the multipliers and MACs as saves the pre-computed possible combinations of the coefficients. Table shows a comparison of multipliers and MACs used in modified DA architecture and conventional implementation. These saved expensive resources can now be utilized in the components where an alternate architecture is not feasible.

MACs and Multipliers		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	0	16
MDFE	0	18
LE	0	16
SDFE	0	8
Precoding in OFDM	0	5

5.2 ROMs

DA architecture saves all the possible combinations of the coefficients of FIR filter. The bits of the input combine and give the address of the ROM containing the appropriate combination of coefficients. Doing this eliminates the multipliers and MACs from the algorithm, at the cost of the ROMs used in the design. Following table shows the comparison of ROMs used for conventional and DA based architecture. ROMs are absent in the conventional implementation as there are no values to be saved previously on FPGA, all the calculations need to be completed on arrival of input sample and the combinational cloud increases. ROMs convert the combinational computation cloud into sequential selection of partial product, saved prior to the arrival of input sample.

ROMs		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	6	0
MDFE	16	0
LE	4	0
SDFE	4	0
Precoding in OFDM	3	0

5.3 Adders

Adders are also a very valuable resource on FPGA. In case of conventional implementation, each multiplier generates large number of partial products which need to be added using adders on the FPGA and increase the resource utilization. In case of the DA architecture, the number of adders is also reduced as we have eliminated the multiplication and MACs. Now there is only limited number of the adders required and that can also be designed based on the requirement or the availability of resources. Following table shows the utilization of the conventional implementation in comparison to the DA based architecture implementation.

Adders		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	12	17
MDFE	12	92
LE	8	4
SDFE	8	12
Precoding in OFDM	8	8

5.4 DSP 48 Blocks

DSP48 block is very expensive and scarce resource on FPGA platform. It consists of an adder and a multiplier combination which perform MAC operation. By eliminating the DSP 48blocks from algorithm, DA architecture minimizes the resource utilization and also improves the frequency of the algorithm. These DSP48 blocks can be used in places where MAC operation must be implemented using multipliers.

DSP48 Blocks		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	0	17
MDFE	0	4
LE	0	20
SDFE	0	8
Precoding in OFDM	0	0

5.5 Registers and Slice LUTs

To save and shift the input, LUT pre-computed entries as well as partial product handling is done in LUT Slices and registers so more resources are required. Minimum number of LUTs and register in the architecture are used through DA based architecture.

Registers		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	352	320
MDFE	132	132
LE	35	7
SDFE	52	24
Precoding in OFDM	56	33
Slice LUT		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	276	1
MDFE	401	262
LE	228	145
SDFE	58	32
Precoding in OFDM	61	49

5.6 Maximum Operating Frequency

The DA based architecture eliminates multipliers, reduced adders. These operations combine in huge quantities in conventional implementation. DA based architecture breaks the combinational cloud into a faster selection method by using the pre-computed coefficient combinations in the LUT. This saves resources and algorithm execution speed also increases.

Maximum Operating Frequency (MHz)		
Filters	DA Based Implementation	Conventional Implementation
FIR Filter	124	37
MDFE	325	199
LE	326	114
SDFE	404	268
Precoding in OFDM	483	249

Chapter 6

Conclusion and Future Work

This chapter states the conclusion of this thesis. Further an insight to the possible future dimensions is suggested in which this research can be extended.

6.1 Conclusion

The Distributed Arithmetic DA architecture solves one of the biggest issues in the implementation of multiplication operation in scenarios where one of the operands is a constant and usually input is in a serial fashion. The design for DA Architecture for FIR filter, Single Carrier and Multi Carrier communication system is discussed in detail including advancements introduced by various engineers time to time. Results for successful implementation of these designs are also described. The DA architecture ensures that expensive system resources are utilized very efficiently without compromising on the precision and accuracy of the system. Further, the system is implemented in a 4-way parallel manner which is not at all a complex problem as we can just replicate the same system multiple times. The combinational cloud is reduced by removing the multipliers and decreasing the utilization of adders hence reducing path delays and enhancing the maximum clock frequency. Although it might increases the latency of the system due to the pipeline stage and order of accumulator used, but this is not of much concern as we are getting the advantage of simplicity and fester clocks. In most communication systems, the inputs are in a bit serial fashion and this architecture exploits the same property for higher efficiency and lower power consumption. Adaptive filters whose filter coefficients need to be updated with time can be easily implemented as we only have to push the new partial products into the LUTs. Flexibility is inherited to this system due to which we can adjust the size of the ROMs and proportionally the rest of the hardware is arranged as

per requirement and availability. This provides a trade-off between the speed and the hardware used in any digital processing system.

To get maximum benefit from these advantages, DA based scheme is most likely to be used digital applications and processing applications involving large number of operations running simultaneously on hardware. Taking example of a typical RF transceiver model, it has components like packet start detector, frequency error estimation and correction, timing error estimation and correction, coarse as well as fine synchronization for timing and frequency, high order equalizers and noise removal filters, channel estimation, channel equalization, Fourier and inverse Fourier transform (in case of OFDM system), high speed analog to digital and digital to analog convertors, decimation and interpolation modules, all these involve complex computations in very high frequencies thus require huge resources on FPGA. In such cases, we can use the discussed DA based architecture for inputs in series like in equalizers and filters. This saves the MACs and DSP48s for processes which cannot be solved through alternate architectures.

Similarly in digital image processing applications, multiple high order filters, noise removal modules and interpolation and decimation, equalizers are used. We can use DA based architecture where possible so that modules which have no alternate can use other expensive resources on FPGA.

6.2 Achievements

The achievements so far include the successful implementation of digital FIR filter by using Distributed Arithmetic (DA) architecture. A little modified flavor is introduced to further reduce the resource utilization. This same idea is used in single carrier communication system at receiver side for equalization process and also multi carrier OFDM system for precoding computation. These implementations have been thoroughly tested and no loss of data or precision is observed. The minimization of these resources results in cost minimization and simpler solution.

6.3 Limitations

Distributed Arithmetic (DA) architecture has successfully been used for minimizing resource utilization. The main limitation is this model is that one of the operands has to be of constant value. Also availability of large ROMs on FPGA also limits the implementation of different algorithms using DA technique.

6.4 Future Work

Dedicated FPGA boards can be designed and fabricated which can provide the direct interface for implementing such algorithms and systems having huge dedicated LUTs and ROMs and least Multipliers. Further we can design an application having multiple components each working on different set of inputs. Now we have alternate implementations for all the components in different implementation architectures So, all the parameters can be set as per requirement.

References

1. A. Croisier, D. J. Esteban, M. E. Levilion and V.Rizo, "Digital Filter for PCM Encoded Signals", U.S.Patent No.3,777,130, 1973.

2. S. A. White, “Applications of distributed arithmetic to digital signal processing: A tutorial review,” IEEE ASSP Mag., vol. 6, pp. 4–19, Jul.1989.

3. H. Yoo, D. V. Anderson, "Hardware-Efficient Distributed Arithmetic Architecture For High-order Digital Filters", IEEE International Conference on Acoustics, Speech and Signal Processing, Vol.5,pp: 125-128,March,2005

4. P. Longa, Ali Miri, "Area-Efficient Fir Filter Design on FPGAs using Distributed Arithmetic" IEEE International Symposium on Signal Processing and Information Technology, pp:248-252,2006

5. W. Sen, T. Bin, Z. Jun, “Distributed Arithmetic for FIR Filter design on FPGA”, in proceedings of IEEE international Conference on Communications, Circuits and Systems, Japan, vol. 1, pp: 620-623, 2007

6. J. Park, B. Richards, B. Nikolić, “A 2 Gb/s 5.6 mW Digital LOS/NLOS Equalizer for the 60 GHz Band”, IEEE Journal of Solid State Circuits, vol. 46, November 2011

7. Slimane Ben Slimane, “Reducing the Peak-to-Average Power Ratio of OFDM Signals Through Precoding”, IEEE Transaction on Vehicular Technology, vol. 56, No. 2, March 2007

8. Dr. Shoab Ahmed Khan, “Digital Design of Signal Processing Systems; A Practical Approach”

9. IEEE 802.15WPAN Task Group 3c (TG3c).http://www.ieee802.org/15/pub/TG3c.html

10. Bernard Sklar, “Digital Communications; Fundamentals and Applications”, 2nd Edition

11. S. Hwang, G. Han, S. Kang, J. Kim, "New Distributed Arithmetic Architecture for Low-Power FIR Filter Implementation", IEEE Signal Processing Letters, Vol.11, No5, pp: 463-466,May, 2004

12. Farhana Sheikh, Melinda Miller, Brian Richards1, Dejan Markoviæ, Borivoje Nikoliæ1A 1–190MSample/s 8–64 Tap Energy-Efficient Reconfigurable FIR Filter for Multi-Mode Wireless Communication 2011

13. Shuzo Kato, Hiroshi Harada, Ryuhei Funada, Tuncer Baykas,"Single Carrier Transmission for Multi-Gigabit 60-GHz WPAN Systems 2012

14. David A. Sobel, Robert W. Brodersen, "A 1Gbps Mixed-Signal Analog Front End for a 60GHz Wireless Receiver", 2008

15. Yu Pan and Pramod Kumar Meher, "Efficient Coefficient Partitioning for Decomposed DA-based Inner-Product Computation" IEEE 2011

16. B. Krill, A. Amira, "Efficient Reconfigurable architectures of generic cyclic convolution", IEEE 2011

17. Shunwen Xiao, Yajun Chen, Peng Luo, "The Design of FIR Filter Base on Improved DA Algorithm and its FPGA Implementation", IEEE 2010

18. Pramod Kumar Meher and Sang Yoon Park, "High-Throughput Pipelined Realization of Adaptive FIR Filter Based on Distributed Arithmetic", IEEE 2011

19. Jiaji Wu, Minli Wang, Jechang Jeong, Licheng Jiao," Adaptive Distributed Arithmetic Coding for Lossless compression"IEEE 2010