



DESIGN & IMPLEMENTATION OF LOW POWER & HIGH FAST SUBSCALAR MULTIPLIER

D. GEETHANJALI¹, P. YASODHA DEVI²
1 PG STUDENT 2 ASSISTANT PROFESSOR
ME-VLSI DESIGN

AVS ENGINEERING COLLEGE, TAMILNADU, INDIA

ABSTRACT

Propose a Sub Scalar Computations multiplier that is high speed yet energy efficient. The approach is to round the operands to the nearest exponent of two. This way the computational intensive part of the multiplication is omitted improving speed and energy consumption at the price of a small error. The proposed approach is applicable to both signed and unsigned multiplications. We propose three hardware implementations of the Sub-Scalar Computations multiplier that includes one for the unsigned and two for the signed operations. The efficiency of the proposed multiplier is evaluated by comparing its performance with those of some Sub-Scalar Computations and accurate multipliers using different design parameters. In addition, the efficacy of the proposed Sub-Scalar Computations multiplier is studied in two image processing applications, i.e., image sharpening and smoothing.

Keywords: lower power consumption, XILINX ISE Design, FFT processor and Fourier transform.

1. INTRODUCTION

To develop low power, high speed and area efficient portable electronic design is a very challenging problem for the hardware designers in the current scenario. Mobile phones, smart cards, assistive listening technology such as hearing aids and PDAs are the example of portable consumer electronic products. The main concerns of these products are not only to extend the operating hours of the battery residing in it but also great computational capacity. Low power design can be developing a system level, technology level, architecture level and circuit level. A larger amount of power can be saved if low power design is achieved at system level. A significant amount of power consumption can be reduced at architecture level but at the cost of delay penalty and area overhead. At architecture and system level, parallelism and pipelining are two main techniques used to reduce power and propagation delay. At technology level, power consumption is going to scale down at the same time as the technology is shrinking day by day. Thus, power saving can be achieved by the improvement in fabrication prefecture size, very low voltages, interconnects and insulator with low dielectric constants. At circuit level voltage scaling, threshold voltage, Transistor sizing, network restructuring power down strategies and logic style are used to achieve low power. In addition to this, this technique also contributes in the reduction of propagation delay and area occupancy as well. Digital signal processing (DSP) is an important unit in electronic devices. Digital Signal Processors (DSPs) are used to perform the common operations such as video processing, filtering and fast Fourier transform (FFT). Such modules perform extensive sequence of multiply and accumulate computations. Multiplication is most fundamental operation in digital computer systems and digital signal processors a large number of transistors with high switching transitions is used to perform variety of multiplication operations. Multiplier consumes 30% power and also occupies 46% chip area in 64-point radix-4 pipelined FFT processor.

Therefore, multiplier is most critical, power hungry arithmetic unit that requires more area and computational time various techniques are applied externally and internally in the past, to achieve energy efficient multiplier designs. External techniques are related to the input data characteristics, whereas an internal technique deals with the system, technology, architecture and circuit level. In literature, different tree based multipliers and array based multipliers are discussed extensively. Array based multipliers consumes low power as compared to Sub-Scalar Computations multipliers. In tree based multiplier, additional hardware is requiring to improve the performance, but at the cost of increased layout and parasitic. On the other side, array multiplier has smaller and regular layout. Therefore, array multiplier is a better choice due to its lower power consumption, smaller layout and relatively good performance. Adder is a fundamental unit of the multiplier, thus it has significant feature size, very low voltages, interconnects and insulator with low dielectric constants. At circuit level voltage scaling, threshold voltage, Transistor sizing, network restructuring power down strategies and logic style are used to achieve low power. In addition to this, this technique also contributes in the reduction of propagation delay and area occupancy as well. Digital signal processing (DSP) is an important unit in electronic devices. Digital Signal Processors (DSPs) are used to perform the common operations such as video processing, filtering and fast Fourier transform (FFT). Such modules perform extensive sequence of multiply and accumulate computations. Multiplication is most fundamental operation in digital computer systems and digital signal processors. A large number of transistors with high switching transitions is used to perform variety of multiplication operations. Multiplier consumes 30% power and also occupies 46% chip area in 64-point radix-4 pipelined FFT processor. Therefore, multiplier is most critical, power hungry arithmetic unit that requires more area and computational time. Various techniques are applied externally and internally in the past, to achieve energy efficient multiplier designs. External techniques are related to the input data characteristics, whereas an internal technique deals with the system, technology, architecture and circuit level.

2. POWER DISSIPATION

The sources of power consumption in digital CMOS circuits are static power dissipation and dynamic power. power consumption of digital CMOS circuits]. $2 P_{fc} V I V I V$ Total L DD SC DD leakage DD = + + α (1) Where is switching transition of a clock cycle, is the output capacitance, VDD is the supply voltage and f is the switching frequency which is fixed in many DSP and dedicated applications, Isc is the short circuit current, and leakage I is the leakage current [28]. In the submicron technology, leakage current is a significant contributor of power consumption. Circuit and Electrical and Electronics Engineering: An International Journal (ELELIJ) Vol 5, No 1, February 2016 3 technology level techniques used dual Vt partitioning, multi-threshold CMOS and power gating approach to reduce the leakage power. Some leakage reduction methods have been presented in Power gating approach can reduce both the components of power dissipation up to a good extent. In this system, power gating approach is considering for the power reduction of proposed multiplier. This technique uses sleep transistors to shut down the supply of the selective logic blocks which are not functional during bypassing operation. PMOS transistor acts as a header switch to connect the supply voltage VDD to logic block and NMOS acts as a footer switch to connect the ground to the logic block. The proposed multiplier utilizes power gating approach and used PMOS header switch in place of tri state buffer which are used by the previously discussed multipliers.

3. LITERATURE REVIEW

EFFICIENT FIR FILTER DESIGN USING SUB-SCALAR COMPUTATIONS COMPRESSION

The major important metrics of the digital signal processing circuits are area, power and delay. In many signal processing system significant area, delay and power consumption is due to the multiplication. With advances in technology, many researchers have tried and are trying to design multipliers which offer either of the following design targets – high speed, low power consumption, regularity of layout and less area The Aim of this research is to design a low cost finite impulse response filter using the concept of faithfully rounded truncated multipliers. The optimization of bit width and the hardware resources are done with good accuracy. In direct FIR filter the multiple constant multiplication are implemented using the improved version of truncated multipliers. The most efficient method for designing optimum magnitude FIR filters with arbitrary specifications is the Remez multiple exchange algorithm. Initially Parks McClellan () is used to find the filter order M for the given frequency response. It is an iteration algorithm that accepts filter specifications in terms of passband and stopband frequencies, passband ripple, and stopband attenuation. Remez () is to find the coefficients for the FIR filter of order M. Then, quantize the coefficients with enough bits and generate the set of uniformly quantized coefficients with equal bit width B.

THE EFFICIENT IMPLEMENTATION OF AN ARRAY MULTIPLIER GUOPING WANG INDIANA UNIVERSITY PURDUE UNIVERSITY,

Multiplication is one of the basic and critical operations in the computations. Efficient implementations of multipliers are required in many applications. In this paper, a new implementation of the array multiplier for unsigned numbers is proposed which significantly reduces the silicon area compared to recently published array multiplier while with no penalty of speed and power. The proposed scheme is applicable for VLSI and FPGA application and it can be easily extended to signed number computations.

FIR FILTER DESIGN BASED ON FPGA

Digital filters include infinite impulse response (IIR) digital filter and finite impulse response (FIR) digital filter. As the FIR system have a lot of good features, such as only zeros, the system stability, operation speed quickly, linear phase characteristics and design flexibility, so that FIR has been widely used in the digital audio, image processing, data transmission, biomedical and other areas. FIR filter has a variety of ways to achieve, with the processing of modern electronic technology, taking use of field programmable gate array FPGA for digital signal processing technology has made rapid development, FPGA with high integration, high speed and reliability advantages, FIR filter implementation using FPGA is becoming a trend. This paper studies used FPGA to achieve a 16-order FIR filter design method.

DESIGN & IMPLEMENTATION 8-BIT SUB-SCALAR COMPUTATIONS MULTIPLIER BHUPENDER PRATAP SINGH, RAKESH KUMAR

Many multipliers that are of high-speed, low power and standard in design are used in multiplication process. Numerous efforts have been used to decrease the number of partial products generated in a multiplication process. One of the efforts is Sub-Scalar Computations multiplier. This work aims at designing and implementation of 8-bit Sub-Scalar Computations multiplier using VHDL language. A Sub-Scalar Computations multiplier is an upgraded version of tree based multiplier architecture. Sub-scalar Computations multiplier use carry save addition algorithm to decrease the latency. Speed of Sub-Scalar Computations multiplier can be improved by using compressor methods. By minimizing the number of partial products we used half adder and full adder in 8-bit multiplier. In this Sub-Scalar Computations multiplication process, we also used carry save adder in the last stage to accumulate the last bits. In this work 8*8-bit Sub-Scalar Computations multiplier construction is examined and simulated in XILINX software. To implementation and simulation of 16-bit Sub-Scalar Computations multiplier we used XILINX ISE Design suite 14.6 software. In this 8-bit Sub-Scalar Computations multiplier circuit our main goals are to decrease the area of multiplier circuit and increase the speed of multiplier.

DESIGN AND IMPLEMENTATION OF FIR FILTER BASED ON SUB-SCALAR COMPUTATIONS MULTIPLIER FOR HIGH SPEED AND LOW POWER ANALYSIS G. PREETHI

The most area and power consuming arithmetic operation in high-performance circuits like Finite Impulse Response (FIR), multiplication is one. There are different types of multipliers to reducing the cost and effective parameters in FIR filter design. Among those this paper use truncated multiplier and modified Wallace multiplier in the fir design. The structural adders and delay elements occupies more area and consumes power in this form so it was a reason to forward the proposed method. In prior FIR filters design with low cost effective results will done by the faithfully rounded truncated multipliers with the carry save additions. In MCMAT design the low cost FIR filters within the best area and power results are implement in this paper by using the improved truncated methods. Along with that the proposed method modified Wallace multiplier based fir filter is also designed in this paper to make the fir filter design is suitable for low power applications.

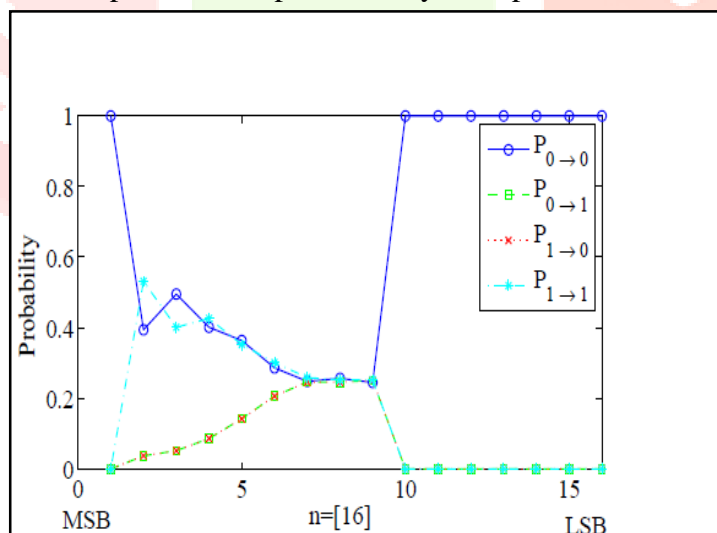
4. EXISTING SYSTEM

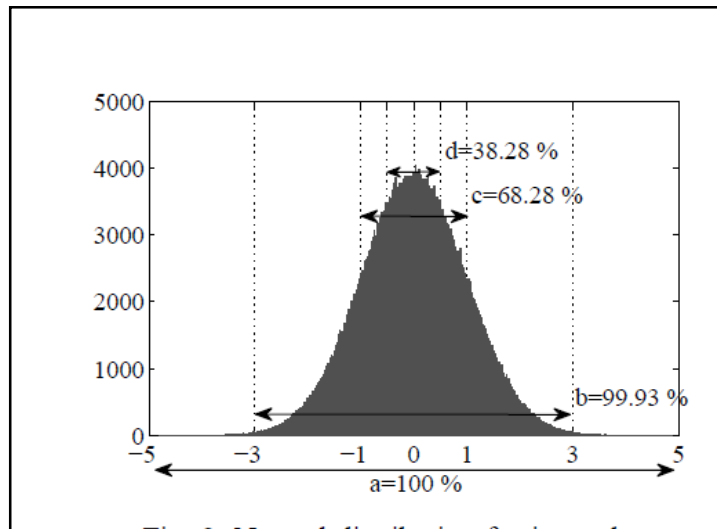
It is a powerful algorithm for signed-number multiplication, which treats both positive and negative numbers uniformly. For the standard add-shift operation, each multiplier bit generates one multiple of the multiplicand to be added to the partial product. If the multiplier is very large, then a large number of multiplicands have to be added. In this case the delay of multiplier is determined mainly by the number of additions to be performed. If there is a way to reduce the number of the additions, the performance will get better. Booth algorithm is a method that will reduce the number of multiplicand multiples. For a given range of numbers to be represented, a higher representation radix leads to fewer digits. Since a k-bit binary

number can be interpreted as $K/2$ -digit radix-4 number, a $K/3$ -digit radix-8 number, and so on, it can deal with more than one bit of the multiplier in each cycle by using high radix multiplication.

5. PROPOSED SYSTEM

Synthesis and implementation tools perform architectural transformations in order to meet the design specifications. For example, retiming, resource sharing, and unused logic removal can be applied during design optimization. Specifically, synthesis tools assume a uniform distribution of the primary inputs of a system and, according to this assumption, can estimate the switching activity of inner nodes and optimize the given designs. Beyond design optimization, exact knowledge of input data sequence is exploited by power simulations for improving the accuracy of the power estimation. Transition or switching probability is the probability of a bit in a word to change value. Four probabilities characterize every bit in terms of switching: denotes the probability that a bit with value x assumes value y . Clearly there exist four transition probabilities and it holds for the 8-bit grey-scale “Lena” image, the probabilities of every bit to change value are depicted in Fig. 1, where the image is input to a multiplier, used to compute the DCT. The other input of the multiplier receives the cosine coefficients. In this set-up, a two’s-complement representation is assumed. The data input is the complete 512×512 image reshaped in frames of 8×8 pixels. The main concept is to develop a general solution for multiplication, applicable to a wider range of applications and not only to a particular algorithm. As a consequence, we do not restrict the word format handled by the multiplier input to the optimized representation, i.e., 8 integral bits, but instead we appropriately map the data format to the 16-bit multiplier. In the 16-bit word used, the 9 most significant bits (MSB) are allocated to the integral part while the remaining bits represent the fractional part. We call this organization (9,7) format. For integral data input, the MSB is always zero due to no negative input. Also, there is no fractional part and thus fractional part and the negative values are used to represent the coefficients assigned to the second input of the multiplier. For the remainder of the bits, the transition probability increases moving from MSB to the least significant bits (LSB), approaching 0.25 for bits 7 through 9, which behave similarly to UWN (Uniform White Noise) Based on this observation, power consumption reduction is sought by using dedicated hardware, which exploits the low switching probability in the MSB part of the input data. Clearly, if the MSB part of the product keeps its value through a sequence of calculations, it would be reasonable not to repeat calculations that depend on this part in every multiplication





In this section the proposed architectures are detailed. Among the various approaches reported in the literature, this paper focuses on minimizing the power consumed in multipliers with the introduction of architectures taking advantage of input data characteristics. Input data characteristics are commonly either not known to the synthesis and optimization tools or cannot be exploited by them for hardware optimization. The basic concept of the proposed architectures is to partition a multiplier in such a way that not all constituent parts are activated for every change of the inputs. For an input sequence that has a normal distribution as the one depicted in Fig. 2, with mean value 0 and variance 1, the dynamic range is assumed to be the interval $(-5\sigma, 5\sigma)$. Table I lists the percentage of input data that fall in several intervals. Data statistics indicate that most inputs take values close to zero. The proposed architectures take advantage of this observation to decrease the power consumption required for multiplication. This can be done by separating the multiplier data path into several independent data paths and deactivating the paths the input of which does not change for a series of input values. The proposed architectures are detailed. Among the various approaches reported in the literature, this paper focuses on minimizing the power consumed in multipliers with the introduction of architectures taking advantage of input data characteristics. Input data characteristics are commonly either not known to the synthesis and optimization tools or cannot be exploited by them for hardware optimization. The basic concept of the proposed architectures is to partition a multiplier in such a way that not all constituent parts are activated for every change of the inputs. For an input sequence that has a normal distribution as the one depicted in with mean value 0 and variance 1, the dynamic range is assumed to be the interval $(-5\sigma, 5\sigma)$. the percentage of input data that fall in several intervals. Data statistics indicate that most inputs take values close to zero. The proposed architectures take advantage of this observation to decrease the power consumption required for multiplication. This can be done by separating the multiplier data path into several independent data paths and deactivating the paths the input of which does not change for a series of input values.

6.1 SELECTIVE ACTIVATION APPROACH

An earlier version of the architecture presented here is reported in. Here we introduce optimizations to further decrease power consumption. The main idea of the proposed

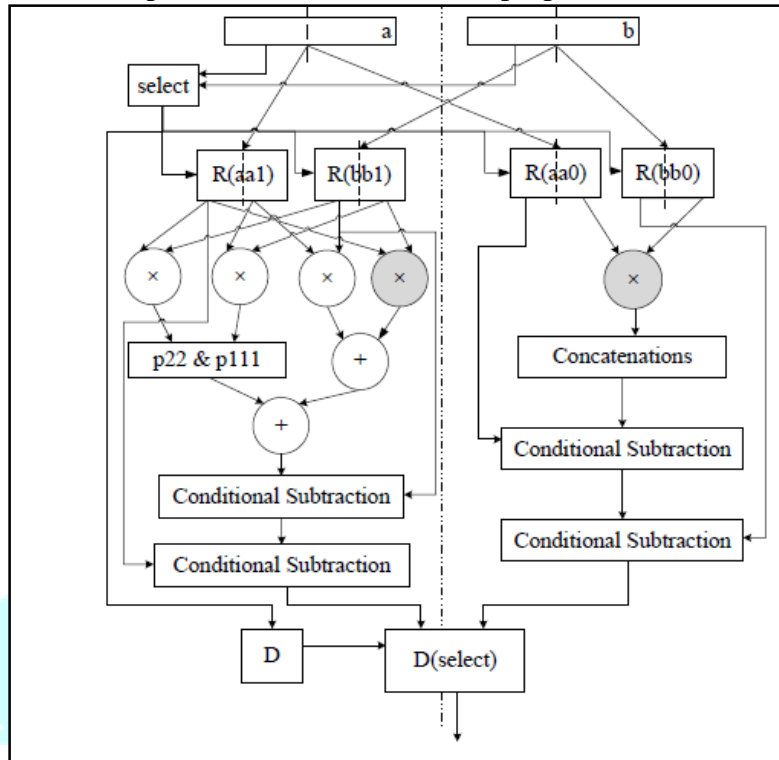


Fig. :1 Selective activation architecture

Selective-activation multipliers are that the architecture provides two paths, one for small number and one for large-number multiplications. When a value is within a predefined range then the path for small-number multiplication is utilized while the other part remains inactive. When a value is outside of the predefined range then the path for large-number multiplication is use. On the left side the large-number multiplication is shown while small-number multiplication is implemented on the right side.

The proposed multiplier can be conceived as a modification of the Baugh-Wooley multiplier. The inputs an and b with bit-width n are split into Ah, Bh and Al, Bl as

$$A_h = a[n-1, m],$$

$$A_l = a[m-1, 0],$$

$$B_h = b[n-1, m],$$

$$B_l = b[m-1, 0]$$

where Al and Bl comprise the m least significant bits of the inputs an and b, while Ah and Bh comprise the n - m most significant bits and m _ n - 2. A select signal is asserted when both Ah and Bh are either all-one or all-zero words, i.e., an and b are within the range [-2m, 2m - 1]. In this case only the small-number path of the architecture is activated which comprises a m × m unsigned multiplier and the necessary sign extension units as it is described in. Concatenations, additions, and subtractions are used in order to calculate the product according to. When an input arrives, it is stored in the corresponding register denoted as R(aa1), R(bb1), R(aa0), R(bb0), depending on the value of the select signal. The selective storage is a difference between the proposed architecture and the earlier version; it guarantees that if a path is not active then all intermediate signals do not change and thus only static power is consumed. The separation between large-number and small-number is clearer. Furthermore, there is one additional multiplier compared to the implementation of in which, the small-number path is embedded into the large-number path. The introduction of the registers forms a pipeline stage. The power consumption decrease achieved reaches 83% and it is much higher compared to in which a 38% power reduction is reported in comparison with the conventional (CSA) multiplier. For an ideal scenario where only the lower l bits change values in a 32-bit multiplier, the corresponding results are reported. which depicts the power consumption as a function of m.? Three experiments are reported. In the first one, only the 8 LSB are excited by a uniformly distributed random input, while the remainder of the word is obtained by sign extension (l = 8). The experiment is repeated for l=16 and l=24. Both proposed and conventional CSA (Carry Save Array) architecture are

measured for power consumption. The reduction of the power consumption achieved is 83%, 65%, and 50% for $l \in \{8, 16, 24\}$ respectively. The optimal selection of the value of m coincides with 1. This is reasonable according to the proposed architecture because the optimal m , which is equal to the value of 1, indicates that minimum power is consumed when the maximum possible number of bits is allocated the high part which does not switch, and only the small multiplication is required for the computation.

6.2 PARTITIONED MULTIPLICATION APPROACH

The selective activation architecture requires that the most significant part of the inputs is all-one or all-zero in order to activate the small-number path. This requirement may not be practical for certain applications. For example, in DWT and DCT, the input data assume a certain value in the most significant part for a sequence of inputs but it is not an all zero value for most of the inputs. To address this issue, a different approach is proposed in the following, based on the partitioned input concept, combined with the minimization of additional circuitry. Algorithm 2 describes the operation of this architecture. Following the concept of partitioned multiplication, the multiplication is divided into four independent multiplications which compute four products p_{22} , p_{21} , p_{12} , and p_{11} . Apart from p_{11} which is unsigned, the remaining three products are signed. The point of partition is determined by m similarly to the multiplication described in section III-A. Each input is stored to one of the corresponding input registers R_{aa0} , R_{aa1} , R_{bb0} and R_{bb1} . Each partial multiplication is executed only when the corresponding input data change value. Based on this concept, only certain parts of the architecture are activated. For example, in image processing using DCT and DWT, the MSB part stays relatively unchanged during processing. Thus, p_{22} requires only static power consumption, while p_{12} , p_{21} exhibit average activity because at least one input remains unchanged, and only p_{11} demonstrates full range power consumption. Additionally, since the input of some multipliers remains unchanged, the partial products remain stable and thus the switching activity induced to the subsequent adder tree is reduced. The proposed multiplier architecture has been found to achieve the lowest power

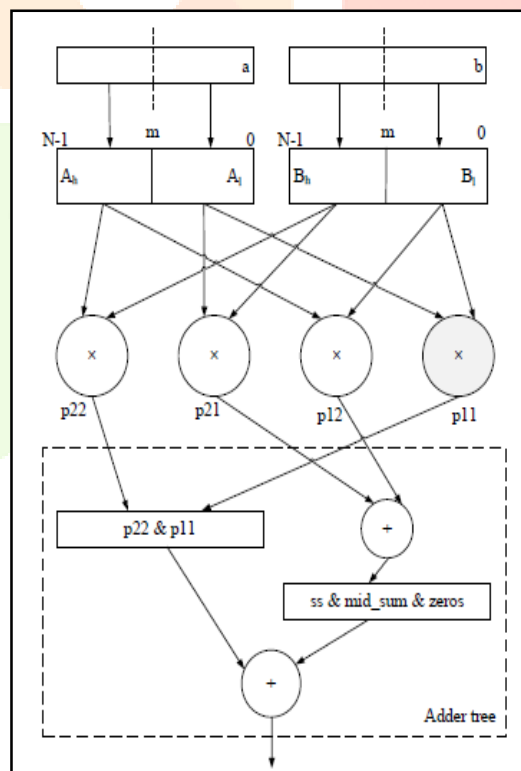


Fig:2 Partitioned Multiplication approach

Consumption for a range of applications and especially for image processing applications such as DCT and DWT.

6.3 AREA AND TIMING REQUIREMENTS

As detailed in previous sections the proposed architectures rely on smaller internal multipliers. In this section the proposed architectures using Carry Save Array (CSA) as the architecture of internal multipliers are compared with conventional CSA multipliers in terms of area and delay. depicts the required area (in μm^2) of the cells used for implementing architectures as a function of n and m , where $16 \leq n \leq 32$ and $1 \leq m \leq n - 2$ for each value of n . The area required by the CSA multiplier depends on n only, as no partitioning is applied. The proposed selective-activation approach requires more area compared to the CSA (11.0%–55.5%). In addition, the partitioning approach requires relatively more area than CSA (1.8%–26.1%), yet clearly less area than the selective activation approach. depicts the maximum required delay of the proposed architectures as a function of n and m , where $16 \leq n \leq 32$ and $1 \leq m \leq n - 2$ for each value of n . The selective activation approach has a shorter critical path compared to CSA multipliers (3.4%–24.6%), achieved by selecting an optimal value for m . In addition, the partitioning approach is relatively closer to the CSA architecture concerning timing requirements (–8.2%–15.2%). depicts the required area for $n = 16$ and for $1 \leq m \leq n - 2$. The selective-activation architecture requires more area than the CSA (23.7%–55.5%), while the partitioned multiplier requires more area than the CSA (8.3% to 26.1%) but less than the selective-activation. In addition, the critical path for $n = 16$ and for $1 \leq m \leq n - 2$ is depicted. The selective-activation multiplier has a 7.4% to 23.5% short critical path, while the critical path of the partitioned multiplier remains almost the same (–7.9%–9.5%). depicts the required area versus delay for $n = 16$ and $m \in \{2, 4, 6, 8, 10, 12, 14\}$. Area-delay points refer to different instances of the architecture obtained by limiting maximum delay to values from 1 ns to 10 ns during synthesis. In the remainder of this paper, all circuits are optimized under a maximum delay constraint of 10 ns.

6.4 FIR FILTER DESIGN USING APPROXIMATION MULTIPLIER

In the field of electronic industry digital filters are used extensively. The noise ranges gradually increase by using analog filters for better noise performance can be obtained by using digital filters compared to analog filters. At every intermediate step in digital filter transformation able to perform noiseless mathematical operations. Our design includes the optimization of bit width and hardware resources without any impact on the frequency response and output signal precision. Addition (or subtraction), Multiplication (normally of a signal by a constant) Time Delay i.e. delaying a digital signal by one or more sample periods are three basic mathematical operations used in digital filters. The coefficients are multiplied by fixed-point constants using additions, subtractions and shifts in a multiplier block. In VLSI Signal Processing two types of digital filters are most widely used one is FIR (finite impulse response) and the other. FIR as indicates that the impulses are finite in this filter and phase is kept linear in order to noise distortions and no feedback is used for such a filters. As compared to FIR is very simple to design. Such type of FIR filters is used in DSP processors for high speed. In Digital Signal Processing Multiplication and addition is of times required. A high speed addition is done by parallel prefix adder and the better version of truncated multiplier with fewer components makes the reduction in delay. In Digital Signal Processing, FIR filters define less number of bits which are designed by using finite precision. In filter by using feedback problems will raise but in FIR filters limited bits are efficient in which there is no feedback. Using fractional arithmetic, we can implement FIR filters. The multipliers play a significant role in arithmetic operations in DSP filtering applications. Multiplication is a hardware intense operation, and the main areas of interest are higher speed, lower cost, and less VLSI area. Two significant yet often conflicting design criteria are power consumption and propagation delay. Considering these constraints, the development of low power multiplier is of great interest. In this paper, we focus on proposing a high-speed low power/energy yet Sub-Scalar Computations multiplier appropriate for error resilient DSP applications. The contributions of this paper can be summarized as follows: 1. presenting a new scheme for APM multiplication by modifying the conventional multiplication approach; 2. Describing three hardware architectures of the proposed Sub-Scalar Computations multiplication scheme for sign and unsigned operations. The proposed FIR filter consists of Sub-Scalar Computations multiplier, which is also area efficient, is constructed by modifying the conventional multiplication approach at the algorithm level assuming rounded input values.

7. SOFTWARE DISCRIPTION

XILINX ISE OVERVIEW

The Integrated Software Environment (ISE™) is the Xilinx® design software suite that allows you to take your design from design entry through Xilinx device programming. The ISE Project Navigator manages and processes your design through the following steps in the ISE design flow.

Design Entry - Design entry is the first step in the ISE design flow. During design entry, you create your source files based on your design objectives. You can create your top-level design file using a Hardware Description Language (HDL), such as VHDL, Verilog, or ABEL, or using a schematic. You can use multiple formats for the lower-level source files in your design.

Note: If you are working with a synthesized EDIF or NGC/NGO file, you can skip design entry and synthesis and start with the implementation process.

Synthesis - After design entry and optional simulation, you run synthesis. During this step, VHDL, Verilog, or mixed language designs become netlist files that are accepted as input to the implementation step.

Implementation - After synthesis, you run design implementation, which converts the logical design into a physical file format that can be downloaded to the selected target device. From Project Navigator, you can run the implementation process in one step, or you can run each of the implementation processes separately. Implementation processes vary depending on whether you are targeting a Field Programmable Gate Array (FPGA) or a Complex Programmable Logic Device (CPLD).

Verification - You can verify the functionality of your design at several points in the design flow. You can use simulator software to verify the functionality and timing of your design or a portion of your design. The simulator interprets VHDL or Verilog code into circuit functionality and displays logical results of the described HDL to determine correct circuit operation. Simulation allows you to create and verify complex functions in a relatively small amount of time. You can also run in-circuit verification after programming your device.

Device Configuration - After generating a programming file, you configure your device. During configuration, you generate configuration files and download the programming files from a host computer to a Xilinx device.

Additional Resources - For additional high-level information, see the following resources:

Project Navigator Overview describes how to process your design using Project Navigator.

FPGA Design Flow Overview describes the steps in the FPGA design flow.

CPLD Design Flow Overview explains the basic steps for processing a CPLD design.

ISE Quick Start Tutorial helps you get started using the ISE software.

ModelSim - Assume that you are familiar with the use of your operating system. You should also be familiar with the window management functions of your graphic interface: Open Windows, OSF/Motif, CDE, KDE, GNOME, or Microsoft Windows 2000/XP. We also assume that you have a working knowledge of the language in which your design and/or test bench is written (i.e., VHDL, Verilog, etc.). Although ModelSim is an excellent tool to use while learning HDL concepts and practices, this document is not written to support that goal. ModelSim is a multi-language HDL simulation environment by Mentor Graphics, for simulation of hardware description languages such as VHDL, Verilog and SystemC, and includes a built-in C debugger. ModelSim can be used independently, or in conjunction with Altera Quartus or Xilinx ISE. Simulation is performed using the graphical user interface (GUI), or automatically using scripts.

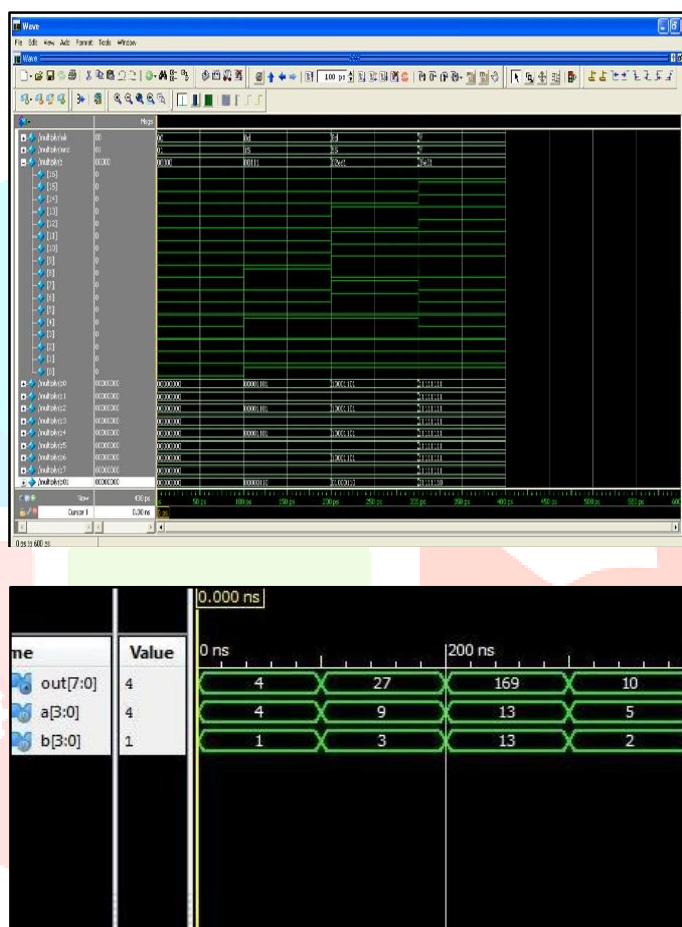
Editions - ModelSim is offered in multiple editions, such as ModelSim PE, ModelSim SE, and ModelSim. ModelSim SE offers high-performance and advanced debugging capabilities, while ModelSim PE is the entry-level simulator for hobbyists and students. ModelSim SE is used in large multi-million gate designs, and is supported on Microsoft Windows and Linux, in 32-bit and 64-bit architectures. ModelSim XE stands for Xilinx Edition, and is specially designed for integration with Xilinx ISE. ModelSim XE enables testing of HDL programs written for Xilinx Virtex/Spartan series FPGA's without needed physical

hardware. ModelSim can also be used with MATLAB/Simulink, using Link for ModelSim. Link for ModelSim is a fast bidirectional co-simulation interface between Simulink and ModelSim. For such designs, MATLAB provides a numerical simulation toolset, while ModelSim provides tools to verify the hardware implementation & timing characteristics of the design.

Language support - ModelSim uses a unified kernel for simulation of all supported languages, and the method of debugging embedded C code is the same as VHDL or Verilog. ModelSim enables simulation, verification and debugging for the following languages:

- VHDL
- Verilog
- Verilog 2001
- SystemVerilog
- PSLSystemC

8. SIMULATION RESULT



9. CONCLUSION

Multiplier is a fundamental component for digital signal processing (DSP) applications and takes up the most part of the resource utilization, namely power and area. Sub-scalar Computations circuitry architectures have been studied as innovative paradigm for reducing resource utilization for DSP systems. In this paper, the 4:2 compressor based Sub-Scalar Computations multiplier architecture which uses both truncation and approximation of compressor is studied. A greedy selection algorithm is then proposed to identify the Pareto frontier to give the optimal accuracy-power trade-off. A finite impulse response (FIR) filter is used as an assessment. The architecture proposed in this paper has achieved up to 21.03% and 27.72% saving on power and area for FIR filter case compared to conventional multiplier designs with a decrease of 0.3dB in output SNR.

10.REFERENCE

- 1.W. D. Hillis, “The connection machine,” Ph.D. dissertation, Dept. Elect. Eng. Computer. Sci., MIT, Cambridge, MA, USA, 1985.
- 2.K. S. Pandey, D. Kumar, N. Goel, and H. Shrimali, “An ultra-fast parallel prefix adder,” in Proc. IEEE 26th Symp. Computer. Arithmetic (ARITH), Jun. 2019, pp. 125–134.
- 3.I. Koren, Computer Arithmetic Algorithms, 2nd ed. Natick, MA, USA: A. K. Peters, 2001. B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs. New York, NY, USA: Oxford Univ. Press, 2000
- 4.M. D. Ergecovac and T. Lang, Digital Arithmetic. New York, NY, USA: Morgan Kaufmann, 2003.

