IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

A LOW POWER AND HIGH ACCURACY APPROXIMATE MULTIPLIER WITH RECONFIGURABLE TRUNCATION

MRS. THENMOZHI, M. E[1], A. LAKSHAMANAN[2]
1 ASSISTANT PROFESSOR 2 STUDENT
VLSI-DESIGN - ELECTRICAL COMMUNICATION ENGINEERING
GNANAMANI COLLEGE OF TECHNOLOGY, NAMAKKAL, TAMILNADU, INDIA

ABSTRACT

Multiplication is the basic building block for several DSP processors, Image processing and many other. Over the years the computational complexities of algorithms used in Digital Signal Processors (DSPs) have gradually increased. This requires a parallel array multiplier to achieve high execution speed or to meet the performance demands. A typical implementation of such an array multiplier is Partial product design. Partial product multiplier is a type of parallel array multiplier. The architecture of Partial product multiplier mainly consists of some Carry Save Adders, array of AND gates and one Ripple Carry Adder. In this research work, a new design of Partial Product Multiplier is proposed and this proposed design of multiplier uses a very fast parallel prefix adder (Kogge Stone Adder) in place of Ripple Carry Adder. The architecture of standard Partial Product Multiplier is modified in this work for reducing the delay due to Ripple Carry Adder and performing faster multiplication of two binary numbers. This research also presents a comparative study of FPGA implementation on Spartan2 and Spartartan2E for new multiplier design and standard partial product multiplier. The RTL design of proposed new Partial Product Multiplier and standard partial product multiplier is done using Verilog HDL. The simulation is performed using ModelSim. The Xilinx ISE design tool is used for FPGA implementation. Comparative result shows the modified design is effective when compared in terms of delay with the standard design.

Keywords: Digital Signal Processors (DSPs), Verilog HDL, Design tool and Simulation.

1. INTRODUCTION

Two major concerns in optimization have been delay and area in VLSI circuit design. Multiplier circuit is chosen to achieve reduction in the area and delay of circuits. Multipliers involve generation and addition of partial products. Owing to large number of partial products, the computation time has become high. In order to reduce the computation time, Partial product Approach was developed. Accurate compressors have been used in Partial Product Approach to reduce the number of partial products. Though accurate compressors give error-free results area and delay does not reach the best level of optimization. To achieve better level of optimization, approximate compressors came into existence. These compressors generate approximate results and also they reduce the area and delay. This idea can be utilized in error tolerant applications. In Image Processing, the size can be reduced by using multipliers. Kogge Stone Adder is incorporated in the existing design and implanted in the compressor for improved speed and for better optimization.

This is carried out in simulation with the help of Xilinx software. So, the main objectives are to achieve better optimization in area and delay of the multiplier circuit, to design the 16X16 multiplier using Partial Product Approach and approximate compressor, to obtain the outputs of the design using simulation, to analyze the performance of existing design and proposed design. 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 at 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 applier 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 Partial product 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 applier 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. LITERATURE REVIEW

EFFICIENT FIR FILTER DESIGN USING PARTIAL PRODUCT 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 (UR) 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 modem 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.

4. DESIGN & IMPLEMENTATION 8-BIT PARTIAL PRODUCT 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 Partial product multiplier. This work aims at designing and implementation of 8-bit Partial product multiplier using VHDL language. A Partial product multiplier is an upgraded version of tree based multiplier architecture. Partial product multiplier use carry saves addition algorithm to decrease the latency. Speed of Partial product 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 Partial product multiplication process, we also used carry save adder in the last stage to accumulate the last bits. In this work 8*8-bit Partial product multiplier construction is examined and simulated in XILINX software. To implementation and simulation of 16-bit Partial product multiplier we used XILINX ISE Design suite 14.6 software. In this 8-bit Partial product 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 PARTIAL PRODUCT MULTIPLIER FOR HIGH SPEED AND LOW POWER ANALYSIS G. PREETHI

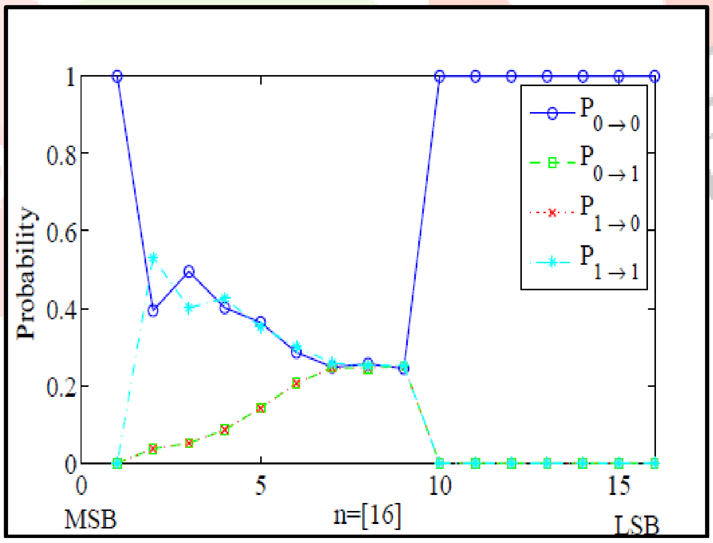
The most area and power consuming arithmetic operation in high-performance circuits like Finite Impulse Response (FIR), multiplication is one. There are different types of multipliers to reducing the cost and effective parameters in FIR filter design. Among those this paper use truncated multiplier and 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.

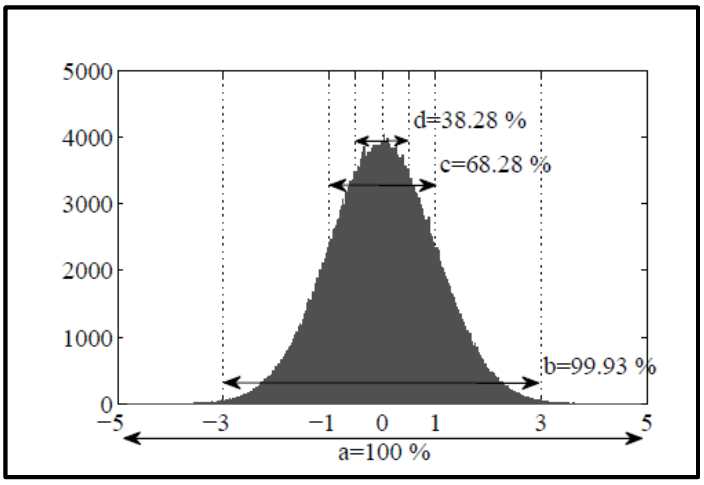
3. EXISTING SYSTEM

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.

4. 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



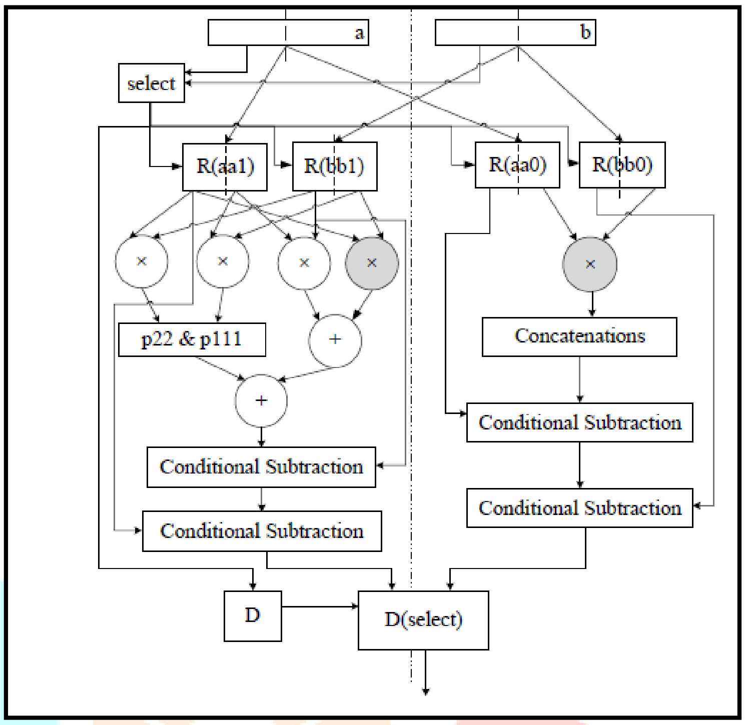


In this section the proposed architectures are detailed. Among the various approaches reported in the literature, this paper focuses on minimizing the power consumed in multipliers
with the introduction of architectures taking advantage of input data characteristics. Input data characteristics are commonly 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 with mean value 0 and variance 1, the dynamic range is assumed to be the interval (−5σ, 5σ). 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 (−5σ, 5σ). 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.

4.1 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



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.

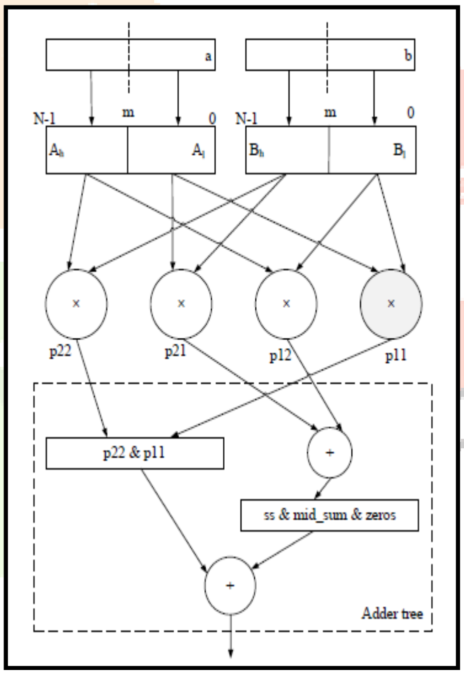
The inputs a and b with bit-width n are split into Ah, Bh and Al, Bl as

$$Ah = a\,[n-1, m],$$
$$Al = a\,[m-1, 0],$$
$$Bh = b\,[n-1, m],$$
$$Bl = b\,[m-1, 0]$$

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 2 {8, 16, 24} respectively. the optimal selection of the value of m coincides with l. This is reasonable according to the proposed architecture because the optimal m, which is equal to the value of l, 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.

4.2 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 p22, p21, p12, and p11. Apart from p11 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 Raa0, Raa1, Rbb0 and Rbb1. 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, p22 requires only static power consumption, while p12, p21 exhibit average activity because at least one input remains unchanged, and only p11 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



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

4.3 AREA AND TIMING REQUIREMENTS

As detailed in previous sections the proposed architectures rely on smaller internal multiplies. 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 µm2) of the cells used for implementing architectures as a function of n and m, where 16 _ n _ 32 and 1 _ m _ 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 _ n _ 32 and 1 _ m _ 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 _ m _ 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 _ m _ 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 2 {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.

5. 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 partial product 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 partial product multiplication scheme for sign and unsigned operations. The proposed FIR filter consists of partial product multiplier, which is also area efficient, is constructed by modifying the conventional multiplication approach at the algorithm level assuming rounded input values.

6. Kogge–Stone adder

In computing, the **Kogge–Stone adder** (**KSA** or **KS**) is a parallel prefix form carry look-ahead adder. Other parallel prefix adders (PPA) include the Brent–Kung adder (BKA),[1] the Han–Carlson adder (HCA),[2][3] and the fastest known variation, the Lynch–Swartzlander spanning tree adder (STA).[4][5]

The **Kogge–Stone adder** takes more area to implement than the Brent–Kung adder, but has a lower fan-out at each stage, which increases performance for typical CMOS process nodes. However, wiring congestion is often a problem for Kogge–Stone adders. The Lynch–Swartzlander design is smaller, has lower fan-out, and does not suffer from wiring congestion; however to be used the process node must support Manchester carry chain implementations. The general problem of optimizing parallel prefix adders is identical to the variable block size, multi-level, carry-skip adder optimization problem, a solution of which is found in Thomas Lynch's thesis of 1996.[5]

An example of a 4-bit Kogge–Stone adder is shown in the diagram. Each vertical stage produces a "propagate" and a "generate" bit, as shown. The culminating generate bits (the carries) are produced in the last stage (vertically), and these bits are XOR'd with the initial propagate after the input (the red boxes) to produce the sum bits. E.g., the first (least-significant) sum bit is calculated by XORing the propagate in the farthest-right red box (a "1") with the carry-in (a "0"), producing a "1". The second bit is calculated by XORing the propagate in second box from the right (a "0") with C0 (a "0"), producing a "0". The Kogge–Stone adder concept was developed by Peter M. Kogge and Harold S. Stone, who published it in a seminal 1973 paper titled A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations.[6]

Enhancements to the original implementation include increasing the radix and sparsity of the adder. The radix of the adder refers to how many results from the previous level of computation are used to generate the next one. The original implementation uses radix-2, although it's possible to create radix-4 and higher. Doing so increases the power and delay of each stage, but reduces the number of required stages. In the so-called **sparse Kogge–Stone adder** (**SKA**) the sparsity of the adder refers to how many carry bits are generated by the carry-tree. Generating every carry bit is called sparsity-1, whereas generating every other is sparsity-2 and every fourth is sparsity-4. The resulting carries are then used as the carry-in inputs for much shorter ripple carry adders or some other adder design, which generates the final sum bits. Increasing sparsity reduces the total needed computation and can reduce the amount of routing congestion.

7. 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. Partial product circuitry architectures have been studied as innovative paradigm for reducing resource utilization for DSP systems. In this paper, the 4:2 compressor based partial product 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. Hence, we can say that for the multiplier applications requiring inputs more than 4 bits, the proposed new design of Braun Multiplier using Kogge Stone adder would provide the result in effectively lesser time than compared with Ripple Carry Adder. 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.

8.REFERENCES

1. M. de la Guia Solaz, W. Han, and R. Conway, ‘‘A flexible low power DSP with a programmable truncated multiplier,’’ *IEEE Trans. Circuits Syst.*, vol. 59, no. 11, pp. 2555–2568, Nov. 2012.
2. A. Momeni, J. Han, P. Montuschi, and F. Lombardi, ‘‘Design and analysis of approximate compressors for multiplication,’’ *IEEE Trans. Comput.*, vol. 64, no. 4, pp. 984–994, Apr. 2015.
3. F. Sabetzadeh, M. H. Moaiyeri, and M. Ahmadinejad, ‘‘A majority-based imprecise multiplier for ultra-efficient approximate image multiplication,’’ *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 11, pp. 4200–4208, Nov. 2019.
4.. Hammad, L. Li, K. El-Sankary, and W. M. Snelgrove, ‘‘CNN inference using a pre-processing precision controller and approximate multipliers with various precisions,’’ *IEEE Access*, vol. 9, pp. 7220–7232, 2021.
5. C. Guo, L. Zhang, X. Zhou, W. Qian, and C. Zhuo, ‘‘A reconfig- urable approximate multiplier for quantized CNN applications,’’ in *Proc.25th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2020, pp. 235–240.
6. B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, ‘‘Quantization and training of neural networks for efficient integer-arithmetic-only inference,’’ in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 2704–2713.
7. P.-Y. Chen, F.-Y. Gu, Y.-H. Huang, and I.-C. Lin, ‘‘WRAP: Weight RemAp- ping and processing in RRAM-based neural network accelerators consider- ing thermal effect,’’ in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2022, pp. 1245–1250.
8. D. Mohapatra, V. K. Chippa, A. Raghunathan, and K. Roy, ‘‘Design of voltage-scalable meta-functions for approximate computing,’’ in *Proc. Design, Autom. Test Eur.*, Mar. 2011, pp. 1–6.-
9. C. S. Wallace, ‘‘A suggestion for a fast multiplier,’’ *IEEE Trans. Electron. Compute.*, vol. EC-13, no. 1, pp. 14–17, Feb. 1964.
10. Z. Yang, J. Han, and F. Lombardi, ‘‘Approximate compressors for error- resilient multiplier design,’’ in *Proc. IEEE Int. Symp. Defect Fault Toler- ance VLSI Nanotechnol. Syst. (DFTS)*, Oct. 2015, pp. 183–186.