A Reliable Low Power Multiplier Using Fixed Width Scalable Approximation

*Stella K*, Vinith T, Sriram K and Vignesh P

1, 2, 3, 4 Vel tech High tech Dr. Rangarajan Dr. Sakunthala Engineering College, Chennai, India

E-mail: 1stellajvk@yahoo.com, 2vinith112000@gmail.com, 3srikaruna.ss50@gmail.com, 4vignesh13999@gmail.com

Abstract. Recent Approximate computing is a change in perspective in energy-effective frameworks plan and activity, in light of the possibility that we are upsetting PC frameworks effectiveness by requesting a lot of precision from them. Curiously, enormous number of utilization areas, like DSP, insights, and AI. Surmised figuring is appropriate for proficient information handling and mistake strong applications, for example, sign and picture preparing, PC vision, AI, information mining and so forth Inexact registering circuits are considered as a promising answer for lessen the force utilization in inserted information preparing. This paper proposes a FPGA execution for a rough multiplier dependent on specific partial part-based truncation multiplier circuits. The presentation of the proposed multiplier is assessed by contrasting the force utilization, the precision of calculation, and the time delay with those of a rough multiplier dependent on specific calculation introduced. The estimated configuration acquired energy effective mode with satisfactory precision. When contrasted with ordinary direct truncation proposed model fundamentally impacts the presentation. Thusly, this novel energy proficient adjusting based inexact multiplier design outflanked another cutthroat model.

Keywords: Low Power Multiplier, Power Dissipation, Speed, Adder

1. Introduction

A multiplier is a hardware block Digital Signal Processing system. In Digital filtering, Digital communications and spectral analysis multiplier plays an important role. For primary design constraints DSP applications target portable, batter operated systems. Reducing the delay of multiplier is essential. Multiplications are very expensive. Two unsigned [1] binary numbers such as X and Y are considered as M and N bits wide. X and Y bits are expressed in binary representation useful for multiplication operation. Using single two input adder is simple form. Using N-bit adder, the multiplication tasks of M cycles are performed by using shift- and -add algorithm [2] which performs addition of M products together. The multiplication operation is generated by the multiplier and in AND operation, the result is shifted based on the position of bits. In long hand decimal multiplication, binary multiplication involves the addition of shifted versions of the multiplicand on value and position of the multiplier bits. Depending on the value of the multiplier bit, the partial products can only be a copy of the multiplier or 0. This is an AND function. Multiplication is similar to manually computing a multiplication. A multi operand and addition is applied to compute the final product. The result structure is called an Array
multiplier and combines the functions, partial product generation, partial product accumulation and final addition. In [3], an approximate procedure is afforded to diminish the quantity of incomplete items. In this technique, input operands calculated approximately by shortening to t and also to h bits as indicated by the situation for driving the slightest bit. These shortened qualities are utilized for expansion and duplication tasks. Likewise, to decrease the mistake coming about because of the truncation activity, we track down the inexact measure of the shortened qualities by adjusting them. These improvements bring about higher exactness and execution contrasted with those of the best-in-class rough multipliers. Additionally, the surmised multiplier has typical blunder circulation with close to zero mean worth. The computation centre of the proposed multiplier performs increase and expansion procedure on shortened and adjusted and outcome is moved to one side to produce the last yield. Since the math activities are performed on the shortened qualities, the estimation centre of this multiplier is little and devours less energy contrasted with the specific multiplier. Additionally, this technique is essentially subject to t and h boundary esteems and isn't altogether influenced by the width of the information operands. This gives an adaptability highlight to the proposed multiplier.

2. Research Background
It is difficult to layout the tree structure which is outlined by Wallace abides due to irregular interconnections. Based upon the binary trees a more uniform tree structure which is represented by [3] and [4]. A binary tree constructs a rule of 4-2 counters, acquires 4 numbers which are added to yield 2 numbers. The final term which are produced by a carry to propagate reduced eight incomplete products into 2 levels which consist of 4 – 2 numbers. At the various places in the reduction the shifting of the partial product introduce zeros Hardware in efficiency is represented by these zeros, actually added zero or from the submission of the irregularities in that tree that Built to explicitly exclude the zeros. The grey dots show beats that jump levels the second level of counter 12 shows more counters in the row then the rows are constructed from first level counters [5]. The irregularities in the layout are all of these effects it is quite more structured than the Wallace tree. The illustration off door diagram is the partial product generation process. For a single bit all zero or one each dot in the diagram place holders. The horizontal row of dots represents the partial products and the selection method. In the upper left corner, each incomplete term is indicated by the table. The arithmetic vertical weight shifts to consider for varying arithmetic calculation of bits in the multiplier by the partial products. At the bottom the final term is exhibited by the double length row.

To sum the partial product and form the final term, the incomplete portion of dot diagram corresponding to the number of hardware essential. If the number of dots present in the vertical column is maximum, then height is represented by latency of that method. This particular linkage varies from logarithmic to linear where linkage delays are important. Adding fewer partial product is good in case of independent of the implementation. The Partial product selection table deduces the logic which selects the partial products. To sum up the partial product the logic can be combined together directly regardless of the hardware used. This selection element can be neglected if the merging reduces the delay of logic elements. The implementation can still be interconnected to the delay because of the physical separation of each AND Gare and of the selection element. For high performance multiplier [6] design, Fast carry propagate adders are important in two ways. Partial Product generation needed “hard” multiples that wanted to be worked as an efficient and fast adder.

3. Proposed System
In Fig.1 we are going to take two 16bit inputs and covert them to fractional part using leading one bit positioning and going to take a 4bit approximation. After the result of the 4bit inputs we are going to add all the approximated values. Finally shifting the result will provide the end results.
Every integer \( N \) can be denoted by
\[
N = \sum_{i=0}^{k} 2^i x_i.
\] (1)
here \( k \) represents the place of leading bit and \( x_i \) represents the \( i \)th bit.
\[
P \times Q = 2^{kP+kQ} \times XP \times XQ.
\] (2)
\( X_P \) and \( X_Q \) represents the width. The approximate value is calculated from the fractional value of \( X_P \) and \( X_Q \).
\[
P \times Q = 2^{kP+kQ} \times (1 + Y_P + Y_Q + Y_P \times Y_Q).
\] (3)
Here \( K_P \) is the leading one-bit position of \( P \) and \( K_Q \) is the leading one-bit position of \( Q \).

Fig 1 proposed system flow chart.

Fig. 2. Dot diagram to assume \( t=7 \) and \( h = 3 \).

Fig. 2. Shows the dot diagram of term \( 1 + (Y_P)^t + (Y_Q)^t + (Y_P)APX \times (Y_Q)APX \) where \( t = 7 \) and \( h = 3 \).

Now, the approximate of (3) may be expressed as
\[
P \times Q \approx 2^{kP+kQ} \times (1 + Y_P + Y_Q + (Y_P)APX \times (Y_Q)APX).
\] (4)
\( Y_P \) and \( Y_Q \) bits are reduced to \( t \) bits to increase the speed
\[
P \times Q \approx (P \times Q) APX = 2^{kP+kQ} \times 1 + (Y_P)^t + (Y_Q)^t + (Y_P)APX \times (Y_Q)APX
\] (5)
16bit X Multiply by 16-bit Y

INPUT PARAMETERS (h, t) h- height t – fraction part
Ex: 1011_1000_001 can be represented as 1.011_1000_001 x 2^{10}
where Green – fraction part
0000_0010_0001_0101 x 0001_1010_001 1_1100
\( 2^{15}, 2^{14}, 2^{13}, \ldots, 2^1, 2^0 \)
Check first ‘1’ from MSB and find its binary location
Here for X first ‘1’ comes at \( 2^9 \) \( K_P=9 \)
Here for Y first ‘1’ comes at \( 2^{12} \) \( K_Q=12 \)
Next to KA fractional parts ASSUME \( t=7; h=3; \)
(XA) \( t = 0000101 \) (YA) \( t = 1010001 \)

APX first 3 bits from (XA) \( t \) and pad ‘1’ at LSB side

(XA)APX = 0001 (YA)APX = 1011

Final computation = ((XA)APX) x (YA)APX + 1 = 8-bit output +( (XA)APX) x (YA)APX + 1 + (XA)t + (YA)

\[ = 0000_1011 +0000 \ 1010 \ \text{(pad 0 at LSB side)} + 1010_0010 \ \text{(pad 0 at LSB side)} = 01 \ \text{1011_0111} \]

POST SHIFT >> KP+KQ= (9+12) =21 (011011_0111) \( << 21 \) (already 8-bit fractional part) \( (21-8) = 13 \) times. Final output is 01 1011_0111 0000_0000_0000_0 which is equal to 35,96,288. Exact output is 11 0110 1001 1110 1110 1100 which is equal to 35,79,628. So that the error difference between them is 16,660 about 1% error rate with 99% accuracy. We will discover a connection among \( t \) and \( h \) boundaries to accomplish a practically high exactness an adequate energy and speed are utilized. The proposed duplication method is possible for unsigned operands. One way to track the supreme worth of the info operands utilized for marked multipliers and duplicated the proposed calculation, and indication of the eventual outcome as per the indication of the information operands. Tracking down the specific supreme worth of the information.

4. Software Implementations

A field-programmable entryway exhibit (FPGA) is a reconciled circuit deliberated to be organized by the originator in the wake of assembling — accordingly "field-programmable". The FPGA setup is the greatest part utilized an equipment depiction language (HDL). The application - explicit coordinated circuit (ASIC) charts were recently used to indicate the design. FPGAs can be utilized to carry out any sensible capacity that an ASIC could perform. The capacity to refresh the usefulness subsequent to transportation, fractional re-setup of the bit of the plan and the low non-repeating designing costs comparative with an ASIC plan (notwithstanding the by and large higher unit cost), offer benefits for some applications. Figure 3 depicts the simulated output.

FPGAs contain programmable rationale parts called "rationale blocks", and a progression of reconfigurable interconnects that permit the squares to be "wired together"— fairly like a one-chip programmable breadboard. Rationale squares can be designed to perform complex combinational capacities, or just straightforward rationale entryways like and XOR. In many FPGAs, the rationale obstructs additionally incorporate memory components, which might be basic flip-lemon or more complete squares of memory. The space of field programmable door cluster (FPGA) plan is developing at a fast speed. The increment in the intricacy of the FPGA's design implies that it would now be able to be utilized in definitely a bigger number of uses than previously. The more current FPGAs are guiding
away from the plain vanilla sort "rationale just" design to one with installed devoted squares for specific applications. With such countless decisions accessible, the creator not just needs to acquaint himself with the different models and their qualities, however he likewise needs an approach to rapidly appraise the exhibition of his plan when focused to the various innovations. This paper momentarily traces the most recent contributions from the key FPGA sellers and in its last half examines the significance of utilizing the correct amalgamation instrument to focus on similar plan to these different advancements.

5. Results and Discussion

![Fig.4 Power Consumption Report](image)

From fig.4 the total thermal power consumption is 61.31mW and the core dynamic dissipation is 0.00mW, Core static thermal power dissipation is about 46.15mW and finally the I/O thermal power dissipation is 15.15mW.

![Fig.5 Area Utilization Report](image)

From the fig.5 the total logic elements used are 409/5,136(8%) with 136/ 5,136(3%) dedicated logic registers. The total pins used are 67/183(37%). The virtual and memory pins are zero.

![Fig.6 Performance Report](image)

From the fig. the maximum frequency at which our proposed system runs is 164.58MHz.
Fig. 7. Register Transfer level (VHDL) Schematic View of proposed system

Table 1. Trade off analyses of approximate DCT over DCT over QUARTUS II hardware synthesis using CYCLONE II family

| DCT model                  | Area (LE’s used) | Speed (MHz) | Total power dissipation |
|----------------------------|------------------|-------------|-------------------------|
| Conventional Direct truncation multiplier | 1197             | 109.33 MHz  | 150.10mW                |
| Approximated DCT           | 409              | 164.58 MHz  | 61.31mW                 |

6. Conclusions and Future Scope

This paper shows the area and energy efficiency of multiplier in which the operands are shortened with dissimilar lengths h and t. Then they are estimated to nearest value of odd numbers in order to diminish the actual error. Speed increased by 50%, area decreased by 70% and energy reduced by 60% as the purpose of this multiplier increases the performance and scalability by using FPGA hardware synthesis, the balance stables. The algorithm proves delay, power and efficiency that can be extended into other algorithm for both 16- bit and 32-bit signed and unsigned data. Active partial product rows can be done by investigating rounding pattern.

7. References

[1] Honglan Jiang, Cong liu, “Low power approximate unsigned multipliers with configurable error recovery”, IEEE transactions on circuits and systems, vol. 66, issue. 1, Jan 2019, pp. 189 – 202.
[2] Mottaghi – Dastjerdi M, “A low power low area multiplier based on shift – and – add architecture”, IEEE transactions on very large-scale integration systems, Vol. 17, Issue. 2, Feb 2009, pp. 302 - 306.
[3] Yen – Jeu Chang, Yu - Chang, “A low power radix-4 Booth Multiplier with pre-encoded mechanism”, IEEE access, Vol. 8, pp. 114842 – 114853, June 2020.
[4] H. Jiang, J. Han, “Approximate radix-8 boothmultipliers for low - power and high - performance operation”, IEEE Transactions on computers, vol. 65, no. 8, pp. 2638 –2644, Aug. 2016.
[5] Kvan – Hung Chan, Yuan - sun chy, “A low power multiplier with the spurious power suppression technique”, IEEE transactions on very large-scale integration systems, Vol. 15, Issue. 7, July 2007, pp. 846 – 850.
[6] Anuradha Chathuranga Ranasinghe, “Glitch – optimized circuit blocks for low power high performance Booth multipliers”, IEEE transactions on very large-scale integration systems, Vol. 28, Issue. 9, Sep 2020, pp. 2028 – 2041.
[7] Parham HosseinZadeh Namin, Crystal Roma, “Efficient VLSI implementation of a sequential finite field multiplier using reordered normal basis in Domino logic”, IEEE transactions on very large-scale integration systems, Vol. 26, Issue. 11, Nov 2018, pp. 2542 - 2552.

[8] Ivan padilla – cantoya, “Low power high parallel load resistance current mode Grounded and floating capacitor multiplier”, IEEE transactions on circuits and systems, Vol. 60, Issue. 1, Jan 2013, pp. 16 - 20.

[9] Shoaleh Hashemi Namin, “Low power design for a Digit – serial polynomial basis finite field multiplier using factoring technique”, IEEE transactions on very large-scale integration systems, Vol. 25, Issue. 2, Feb 2017, pp. 441 - 449.

[10] Mohammed Saeed Ansari, Honglan Jiang, “Low power approximate multiplier using encoded partial products and approximate compressors”, IEEE Journal on Emerging and selected topics in circuits and systems, Vol. 8, Issue. 3, Sep 2018, pp. 404 -416.

[11] Weiqiang Liu, Lian – gyu Qian, “Design of approximate Radix – 4 Booth Multipliers for error tolerant computing”, IEEE transactions on Computers, Vol. 66, Issue. 8, Aug 2017, pp. 1435 - 1441.

[12] Weiqiang Liu, Jiahuaxu, “Design and evaluation of approximate logarithmic multipliers for low power error tolerant applications”, IEEE transactions on circuits and systems, Vol. 65, Issue. 9, Sep 2018, pp. 2856 - 2868.