Industrial Engineering Journal
ISSN: 0970-2555
Volume : 52, Issue 7, July : 2023

Design Of Compression Tree Based Wallace Tree Multiplier For HighSpeed VLSI Application

GANGANAMONI ILAIAH
M.Tech student ECE, VLSI system design. Avanthi institute of Engg & Tech.
Dr.S. Kishore Reddy
Associate professor, HOD ECE, VLSI design Avanthi institute of Engg & Tech.
E mail : kishorereddy416@gmail.com
G.srinivas
Assistant professor, ECE Department, Avanthi institute of Engg & Tech.
E Mali: : sri414china@gmail.com

ABSTRACT

The concept of the estimated multiplier is a potential approach for many error-resilient systems to rising energy usage. We suggest an approximately 8 x 8 multiplier configuration with low power accuracy. in this phase. Two key aspects of the new architecture. The first is that various weights use various compressors to collect their component words depending on their value (in specific precision levels). It makes aminor mistake to reduce the power usage. Furthermore, we use rough-order (e.g. the 8:2 compressor) compressors to which the rationale of transport chains with small weights.To our knowledge, the proposed application is the first function in which approximate compressors in approximate multiplier architecture was effectively used. Experimental tests suggest that the estimated multiplier can achieve either low power and high precision relative to an actual multiplier (Dadda tree multiplier).

Key Words: ***multiplier, architecture, approximate compressors***

1. Introduction

One of the most important design requirements for any electronic device is the ability to save energy, and this is particularly true of mobile systems like smartphones, tablets, and other devices. Any such reduction at the cost of just a little time is highly sought for. The most important parts of developing interactive software for such portable instruments are the DSP frames necessary for processing the signals. The functional core of these SPD structures is an arithmetic logic unit wherein multiplication plays a central role in all arithmetic operations. It is also important to improve multipliers' speed and power/energyconsumption qualities for greater CPU performance. Many of the DSP cores include built-in visual and video processing algorithms, with the resulting renderings suitable for either still images or moving ones.

Fast multipliers are crucial to optical signal systems. Since media processing has begun, the speed of multiply operations has been more important in both digital signal processing and general purpose processors. Historically, a series of addition, subtraction, and rearranging steps has been used. The term "multiplication" is often used to describe aseries of consecutive additions. To preserve information, the product is typically twice as lengthy as the operands if they are considered integrals. The numerical idea proposes a repeating kind of inclusion that is too slow to be nearly alwaysreplaced by a conditional layout method, and this is the case.

2. Literature Survey

In this section, we trace the lineage of our proposed approach in approximation computing back to some earlier work based on approximate multiplier architecture. Gupta et al. developed a proposal for an approximate adder that improves on the strength,range, and efficiency of a standard mirror adder by borrowing part of the rationale behindthe design. A bio-inspired approach proposed by Mahdiani et al. involves approximating the inputs' combined effects using OR windows. Estimated adders are presently the subject of new methods and indications for their modelling and estimation. Babic et al. proposed a piped log-based estimate to the form of a conventional Mitchell multiplier to achieve higher accuracy.

An error-tolerant multiplier is defined by dividing the multiplication into accurate (multiplication-based) and approximate (non-multiplication-based) halves. For partial component inclusion with tunable error recovery, Liu et al. proposed employing an estimated multiplier with a severely uncertain replacement. The method proposed by Kulkarny et al., in which an erroneous block is used to estimate a partial product, is then applied to the final result. In this way, a faulty building block is used to produce widespread multipliers.

3. Existting Techniques

3.1 Basic Multiplication Schemes

In comparison to other arithmetic and logical, multiplication equipment also requires a lotof time and energy. A multiplier / MAC machine is used as a simple building block bythe digital signal processors, mostly with multi-intensive algorithms. Two steps may be preceded by a multiplication procedure:

1) Generate the partial products.

2) Accumulate (add) the partial products.

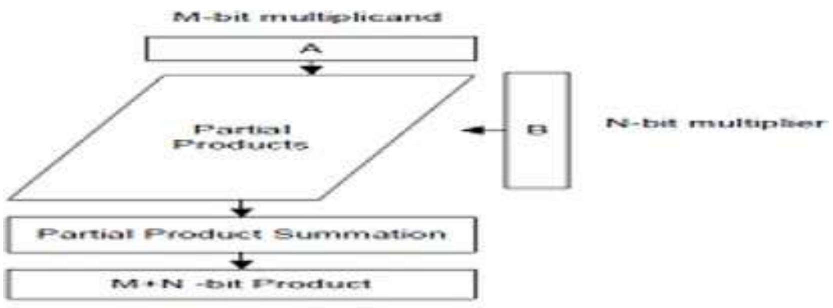


Fig. 3.1: Generic Multiplier Block Diagram

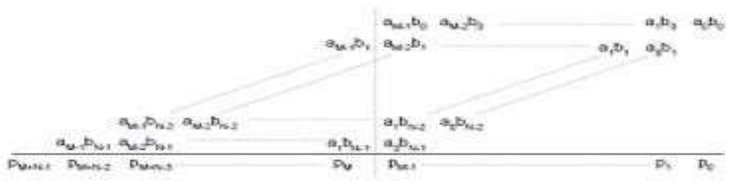


Fig. 3.2: Partial Product Array

3.2 Array Multiplier

The multiplier multiplies each and produces N partial products by something.

That ofthese pieces is either moved by multiplicates or by 0. The partial output production consists of basic multiplier ANDing and multiplicanding.

3.3 Tree Multiplier

When introducing both of them concurrently, the tree multiplier decreases the period togenerate partial items. The multiplier sequence applies each partial platform in order.
CSAs are also used to collect partial goods in the tree multiplier.

3.4 Wallace Tree

Partial items were commonly identified as the Wallace Tree with complete adders as carrion-save adders, often called 3:2 counters. The definition of the 8 * 8-bit partial product trees reduction can be seen in Figure 3.3.

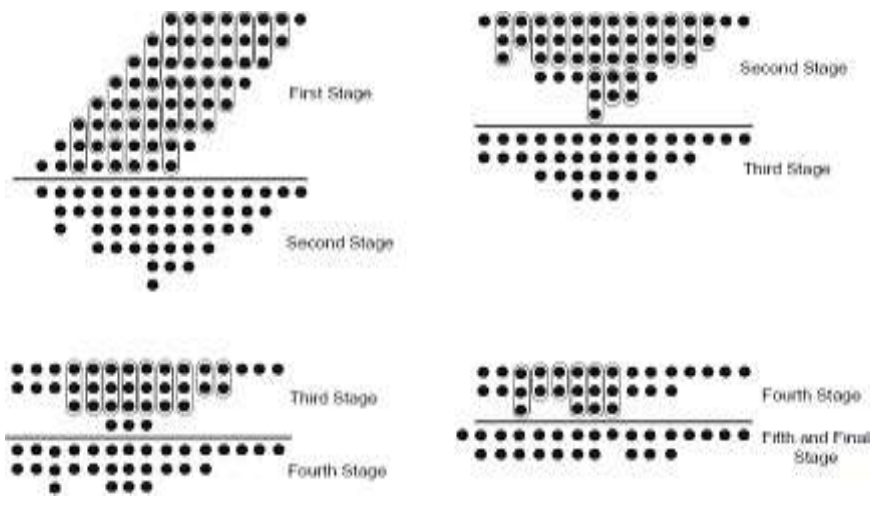


Fig. 3.3: Wallace Tree for an 8 * 8-bit partial product tree

4. Proposed Systems

4.1 Proposed Approximate Multiplier

The work plan of a multiplier is sometimes related to just the maximum height of PPM. So the PPM has to be compressed. A n:2 compressor is a multiplier slice that when properly replicated reduces n (i.e. product terms) to two numbers. The n:2 compressor collects one or more bit of the bottom positions of n:2 in Slice I (e.g. i-1), generates two bits in the I and i+1 positions and one or more bits at the higher positions.

In the traditional design of the multiplier, 4:2 compressors are used[1,2]. The block diagram for an exact (i.e., exact) 4:2 compressor is shown in Figure 4.1(a). The input fourbits are X0, X1 , X2 and X3. The 2 output bits at positions I and i+1 are referred to respectively as Sum and Carry. The carrying bit is called Cin by the lower position, whilethe carrying bit is identified as Cout. The block diagram of an or thereabouts. 4:2 compressor is presented in Figure 4.1(b). The carrying pieces Cin and Cout are removed in order to save the sense of transporting strings. In addition, Sum and Carry logics (e.g., different of Sum and Carry logics in the accurate 4:2 compressor) is redesigned in [1,2] toreduce the error rate. Recent analysis [1,2] neglected to consider compression with high order (i.e., did not take n into account 5). High-order compression can reduce delay and power further. We present our high order design for compressors (that is to say n part 5) in this section.

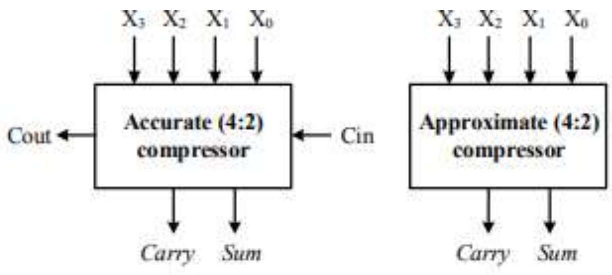


Fig. 4.1: (a) Accurate compressor (b) Approximate compressor

4.2 Proposed Approximate Multiplier Design

For addition, three pieces are a multiplier. For partial goods, AND gates are used in the first component. In the second section, a hold save adder tree lowers the max speed of PPM (partial component matrix). In the third component, the final product is a transportable propagation adder. The multiplier 's architecture sophistication is linked in particular to the PPM reduction circuitry (i.e. it needs to do with the PPM reduction circuitry in particular (i.e. second part). The Multiplier Technology Study[1-6] also concentrates on designing the circuitry for PPM elimination. We propose a configuration of approximately 8 x 8 multipliers in this segment. The configuration of our PPMmitigation circuitry is shown in Figure 4.2.

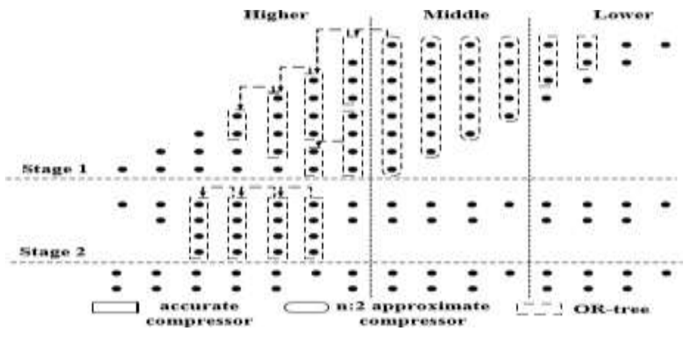


Fig.4.2:PPM Reduction In The Proposed Approximate Multiplier

The weights are categorized according to their definition under three categories: the higher weights, the medium weights and the lower weights. This is important to remember that the designers should specify the number of higher weights, the number of mean weights and the number of shorter intervals for the trade in power usage and numerical precision.

Our PPM reduction circuit utilizes the following sense guided logic compression technologies to minimize power usage with a minor factor: higher weights are using the actual (i.e., actual) of the 4:2 compressors, middle weights are using for our estimated high-order compressors (i.e. the estimated n:2 compressors) and unreliable compressors are using for the lower weights.

There are two phases to our PPM reduction chain. For both weights, the first stage is. Thesecond stage is for weights of greater value only. Increasing weight has a limit of two component terms until the second step is finished. The final result will then be achieved from a carry propagation adder. The specifics of the two phases are listed below.

4.3 Analysis Of Multiplier Using Approximate 4:2 Compressor

Approximate computing has drawn great attention as an attractive worldview for error tolerant applications that are able to accurately reduce energy consumption, delays and area. This article suggests the construction of a modern compressor of approximately 4–

2. A updated Dadda Multiplier design would be introduced to reduce the error on the performance for the effective usage of the proposed compressor.

4.3.1 Exact And Approximate Compressor

The function of the actual 4:2 compressor and the hypothetical compressor suggested is addressed in this section. Four compressors (X1; X2; X3; X4; Cin) and three (sum; carry;cout) are included in the 4:2 compressor. Both inputs are of the same binary weight and output value.

The compressor receives a binary bit less in value from the previous order block and generates Cut and Carry outputs one binary bit more in sense. The same 4:2 compressor circuit diagram is shown in the Fig. 4.6(a)(s). An integrated 4:2 compressor architecture proposed in [9] is used to evaluate estimated compressor output in this article.

$$Sum = X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{in} \tag{4.1}$$

$$C_{out} = (X_1 \oplus X_2)X_3 + \overline{(X_1 \oplus X_2)}X_1 \tag{4.2}$$

$$Carry = (X_1 \oplus X_2 \oplus X_3 \oplus X_4)C_{in} + \overline{(X_1 \oplus X_2 \oplus X_3 \oplus X_4)}X_4 \tag{4.3}$$

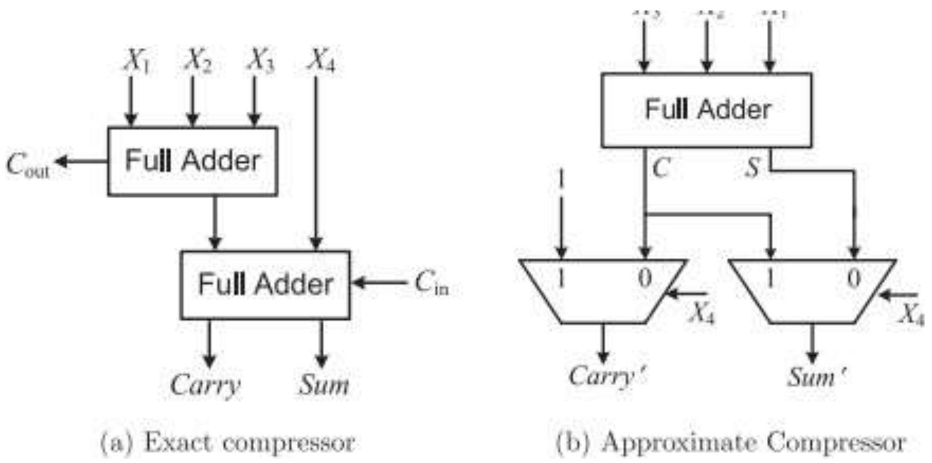


Fig.4.3 :Sum Of 4:2 Compressor A)Exact B) Our Approximate

Around 4:2 is planned with the amount of output bits being decreased to two. Since this estimated compressor is used in the multiplier specification, the decrease of output bits essentially decreases the amount of input bits in the multiplier layout for the succeeding compressors to four. The output bit Cut = 0 in the other 15 instances, with the exception of the input variation of "X4X3X2X1=1111'.' Cout is also not taken into consideration when constructing estimated compressors.

4.3.2Approximate Multiplier

In this section, design of an 8*8 approximate multiplier is presented. The multiplication operation can be divided into 3 steps as follows.

1. Generation of partial products by the multiplication of each bit of multiplicand with each bit of multiplier.
2. Accumulation of partial products into two rows.
3. The computation of final binary result generally using carry propagate adder.

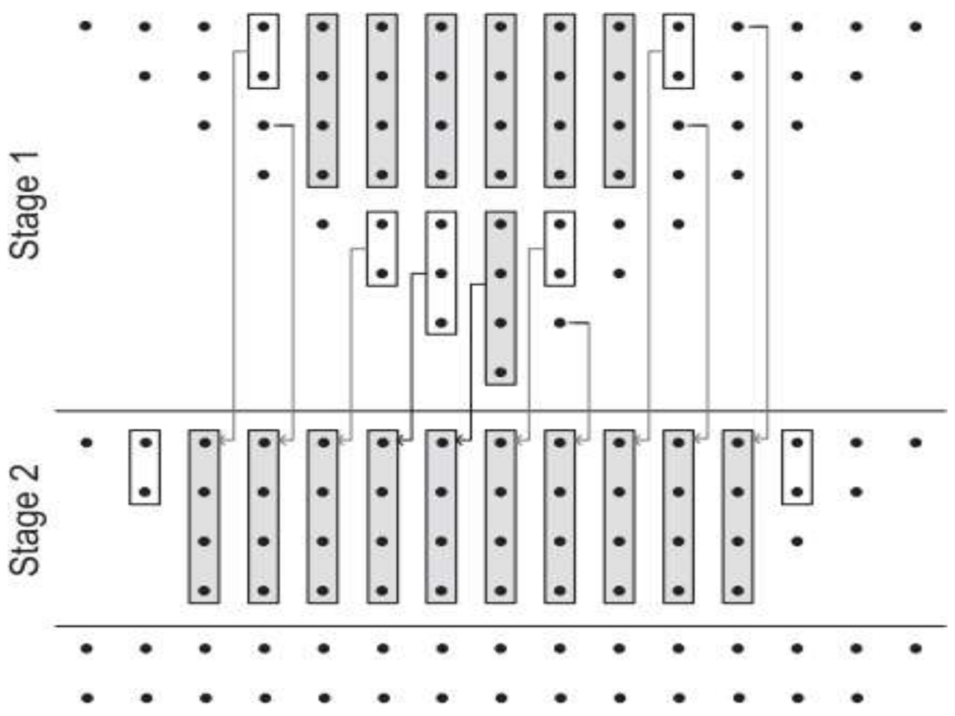
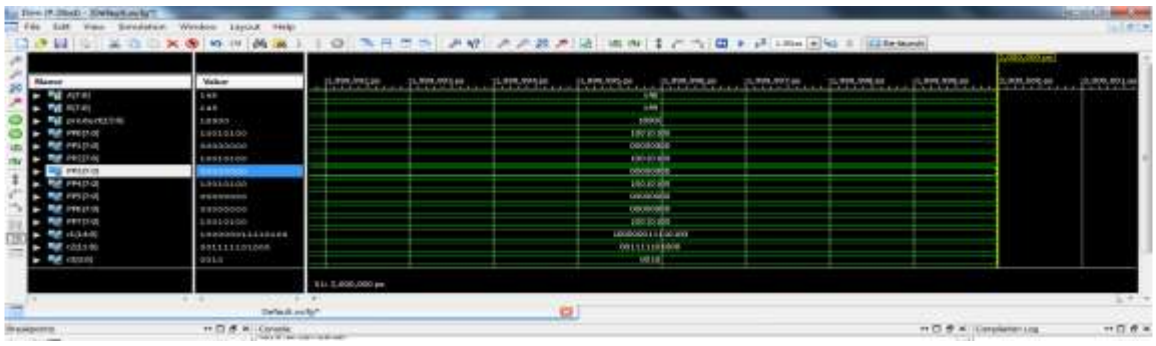


Fig.4.4 :8:8 Approximate Multiplier

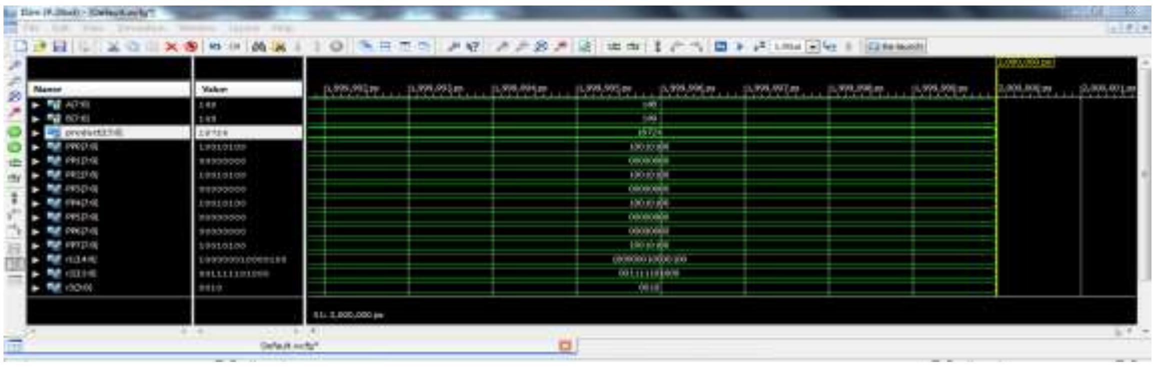
5. Result And Discussions

5.1 Simulation Result

Approximate Sum Approximate Carry



Exact_Sum_Approximate_Carry:



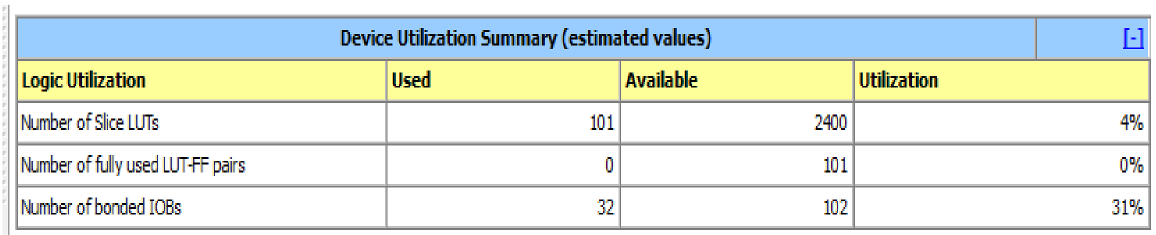
5.2 Synthesis Result

The programs that have been built are simulated and tested. The RTL analysis is carried out using the Xilinx ISE method after practical testing. The RTL pattern, which isassigned to a certain hardware repository, would be translated to the Gate level mesh set.

A number of different tools were included in the Xilinx ISE platform here in this Spartan 3E unit. For this synthesis, the computer called "XC3S500E" and the kit "FG320" have been selected with the pace of the system like "-5." This concept has been synthesized and the effects analyzed:

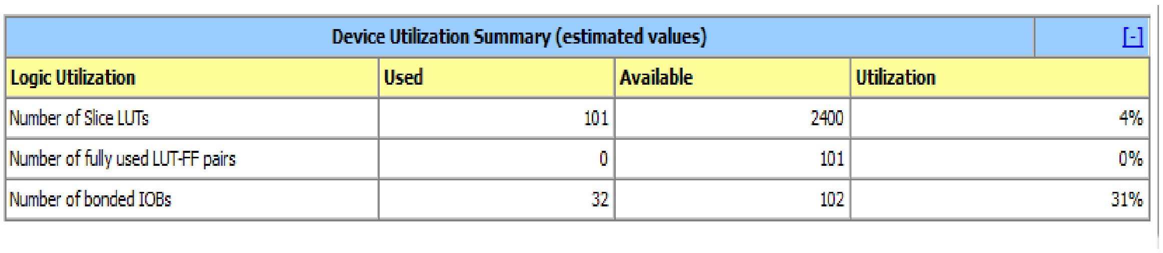
5.3 Design Summary

Approximate Sum Approximate Carry



Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	101	2400	4%
Number of fully used LUT-FF pairs	0	101	0%
Number of bonded IOBs	32	102	31%

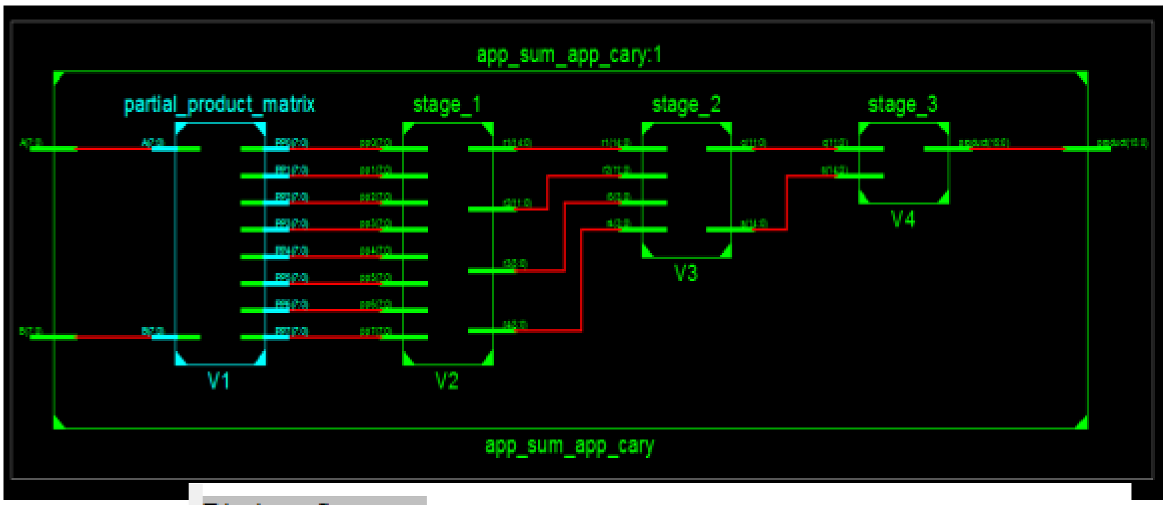
Approximate_Sum_Approximate_Carry:

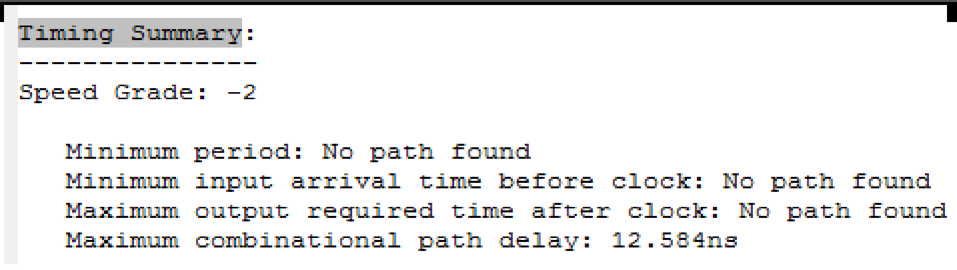


Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice LUTs	101	2400	4%
Number of fully used LUT-FF pairs	0	101	0%
Number of bonded IOBs	32	102	31%

5.4 RTL Schematic

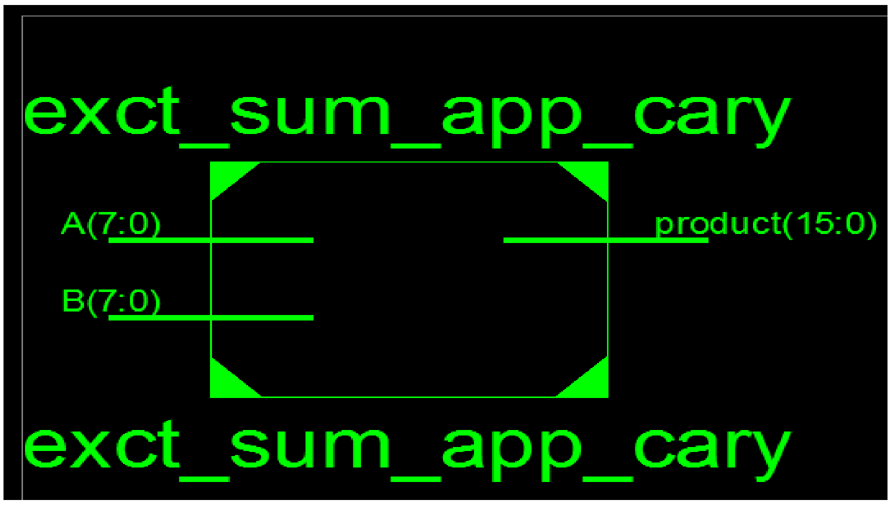
Approximate Sum Approximate Carry





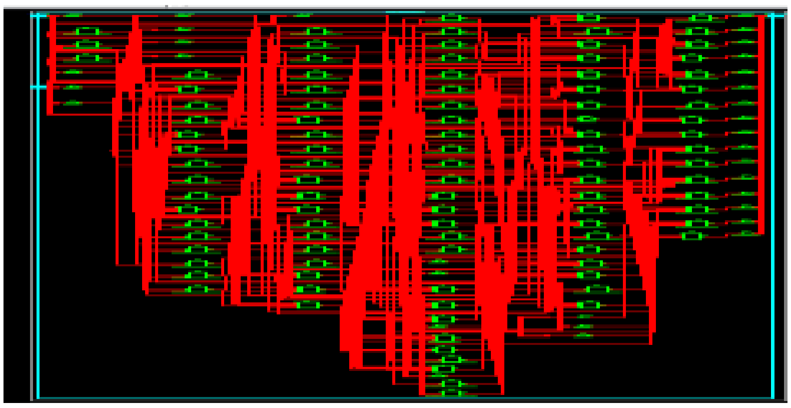
```
Timing Summary:
---------------
Speed Grade: -2

   Minimum period: No path found
   Minimum input arrival time before clock: No path found
   Maximum output required time after clock: No path found
   Maximum combinational path delay: 12.584ns
```

Exact_Sum_Approximate_Carry:

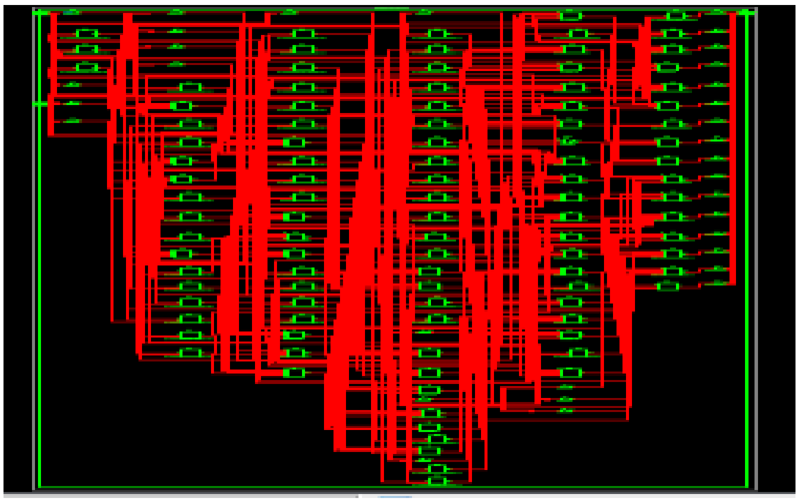


5.5 Technological Schematic

Approximate Sum Approximate Carry



Exact_Sum_Approximate_Carry:



Conclusion

In this post, we have proposed a rough multiplier named RoBA multiplier, which is high-speed but energy efficient. The suggested multiplier was focused upon rounding the inputs in 2n format, which was highly effective. The calculally sensitive portion of the multiplication was therefore skipped for the sake of a small error to increase efficiency and energy usage. The solution suggested extended to both signed and unsigned addition and multiplication. Three device solutions were addressed, including one for the unsignedoperations and two for the subscribing operations. The efficiencies of the recommended multipliers were tested using various architecture criteria in contrast with those of other reliable and rough multipliers. In most (all) instances, the RoBA multiplier architectures surpassed the corresponding (exact) multipliers. In two image processing applications for sharpening and smoothing, the efficacy of the suggested approximate multiplicationmethod was tested. The picture output was similar to that of same algorithms in the contrast.

References:

[1] M. Alioto, ―Ultra-low power VLSI circuit design demystified and explained: A tutorial,‖IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 59, no. 1, pp. 3–29, Jan. 2012.

[2] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, ‒Low-power digital signal processing using approximate adders,‖ IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[3] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, ‒Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications,‖IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.

[4] R. Venkatesan, A. Agarwal, K. Roy, and A. Raghunathan, ‒MACACO: Modeling and analysis of circuits for approximate computing,‖ inProc. Int. Conf. Comput.-Aided Design, Nov. 2011, pp. 667–673.

[5] F. Farshchi, M. S. Abrishami, and S. M. Fakhraie, ‒New approximate multiplier for low power digital signal processing,‖ inProc. 17th Int. Symp. Comput. Archit. Digit.Syst. (CADS), Oct. 2013, pp. 25–30.

[6] P. Kulkarni, P. Gupta, and M. Ercegovac, ‒Trading accuracy for power with an underdesigned multiplier architecture,‖ inProc. 24th Int. Conf. VLSI Design, Jan. 2011, pp. 346–351.

[7] D. R. Kelly, B. J. Phillips, and S. Al-Sarawi, ‒Approximate signed binaryinteger multipliers for arithmetic data value speculation,‖ in Proc. Conf. Design Archit. Signal Image Process., 2009, pp. 97–104.

[8] K. Y. Kyaw, W. L. Goh, and K. S. Yeo, ‒Low-power high-speed multiplier for error- tolerant application,‖ in Proc. IEEE Int. Conf. Electron Devices Solid-State Circuits(EDSSC), Dec. 2010, pp. 1–4.