A Project Report

on

Cooperative Arithmetic Aware Approximation Techniques for Energy Efficient Multipliers

By

Aruna

Shri Lakshmi

Srinivasan M P

Surabhi Mondal

Under the supervision of

Dr. S Ershad Ahmed

Birla Institute of Technology and Science, Pilani - Hyderabad Campus

June 2021

CONTENTS

1. Introduction

The need for approximate computing arises from the fact that sometimes approximate answers are enough to satisfy the requirements, as for instance, a search engine, where we get an approximate search result. Extending the concept to the hardware domain, approximate computing tends to save energy and power by reducing the need to incorporate a huge number of hardware components. This applies especially to power hungry components such as multipliers and adders. But accuracy needs to be traded off to get an energy efficient system. Hence the emergence of arithmetic aware approximation, which introduces some control parameters, which is to say limitations to use this approach, to control the error values generated by this method. This work is basically built on top of three basic approximate techniques, based on prior work, namely pruning, radix encoding, rounding and dynamic scaling.

In Pruning, also called Probabilistic Pruning, certain blocks of circuit along with their associated wires are removed so as to save power, area, and also remove extra overheads, thus reducing time delay as well. However, this is subjected to the extent of error tolerance of the circuit**.** In radix encoding, an approximate number of partial products are generated and then added using compressors. In rounding, the LSBs of the generated partial products are eliminated vertically, and a correction constant is added so as to get rid of the high error obtained in this method. In dynamic scaling, the multiplication is done between m-bit numbers either starting from MSB or ending at LSB. But its usage is limited to small no of input bits.

Based on the above four methods, the implemented paper makes three basic blocks that examines approximation technique :

- High Radix
- Rounding
- Perforation

More specifically, these three modules have been mixed and matched at various lengths to generate output at different degrees of inaccuracy and power consumption.

1.1. Approximate Multiplier Concepts

An inexact multiplier has evolved through various methodologies that aim at contributing less hardware units in implementation. The concepts that have been adopted from literature alleviate the count of partial products generated reflecting its benefits in unit gate area

savings. An equal focus has been given to accuracy and energy consumed by the germane multiplier. An elaborate description on the howabouts of the concepts are expatiate below.

1.2. High Radix

One of the extensive and advantageous gambit conceptualised to the question of framing a notion for reduced partial product generation is performing operations in higher radix. When the multiplier is an n-bit number, operating k bits at a time (k<n, k is a multiple of 2) affirmatively reduces the partial product generation to n/k times. From the implementation point of view, reduced partial product accounts to reduced number of bits in question that need computation. Since the number of bits is proportional to the logical gate units, the structural level of the multiplier is built with abridged complexity.

A high radix in itself has two levels of operation. One of them is the radix encoder and then the partial product generator. A radix 'm' encoder changes the value that it operates on from base 2 to base 'm'. This eventually brings down the count from n bit number to n/m bit number. The encoded values are then evaluated for the output. As the number of the encoded value increases by (logn) time as 'm' varies as it is in range [0, 2^k], the computation phase of the multiplier block requires additional support from the hardware which obverse the prime objective of the concept. To meet the concomitants of the conventional procedure high radix calls for, the range of the encoded value is restrained to [-k,k] for radix-2^k multiplier. The encoded values are represented in powers of 2 in count of k which serves as a control signal that decides the scalar value of multiplicand.

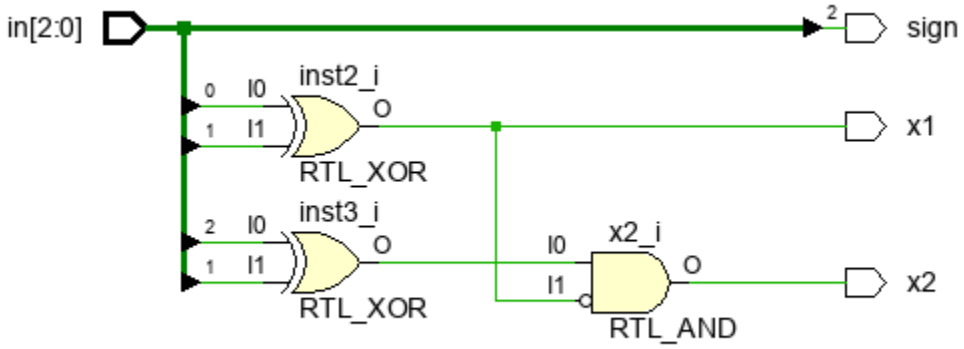


Figure . Radix-4 encoder circuit

A radix-4 encoder has been designed and outputs sign, x1 and x2. Similarly radix-64 encoders output sign, x1, x2, x4, x8, x16 and x32. These outputs serve as control lines of the partial product generator.

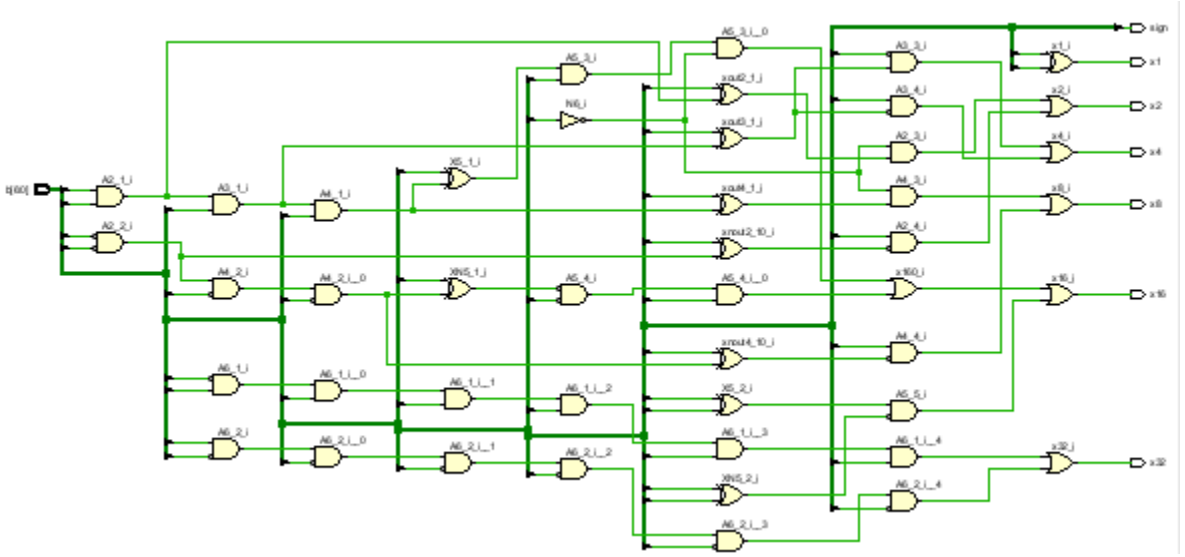


Figure . Radix-2^6encoder circuit

The partial product generator block of high radix uses the inputs of the encoder module and decides the scaling value of the untreated operand. Since k number of elements of 2 are generated for a given high radix, k bits of multiplicand, starting from LSB in overlapping fashion, are treated at a time for a partial product bit. For the partial product generated for -k encoded value, the sign bit has to be added since the module computes the 1's complement of the multiplicand. The bit count of partial products are maintained at n+k to compensate for the shift operation initiated by encoded values.

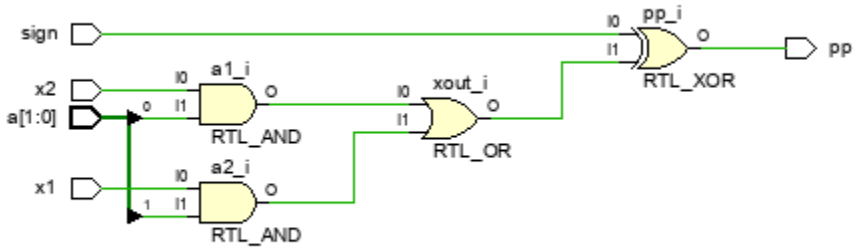


Figure . Radix-4 Partial Product Generator

In radix-4 partial product generator, two inputs of the multiplicand are operated at a time and a single partial product is generated. Similarly, in radix-64 multiplier, 5 inputs are treated for a single partial product bit.

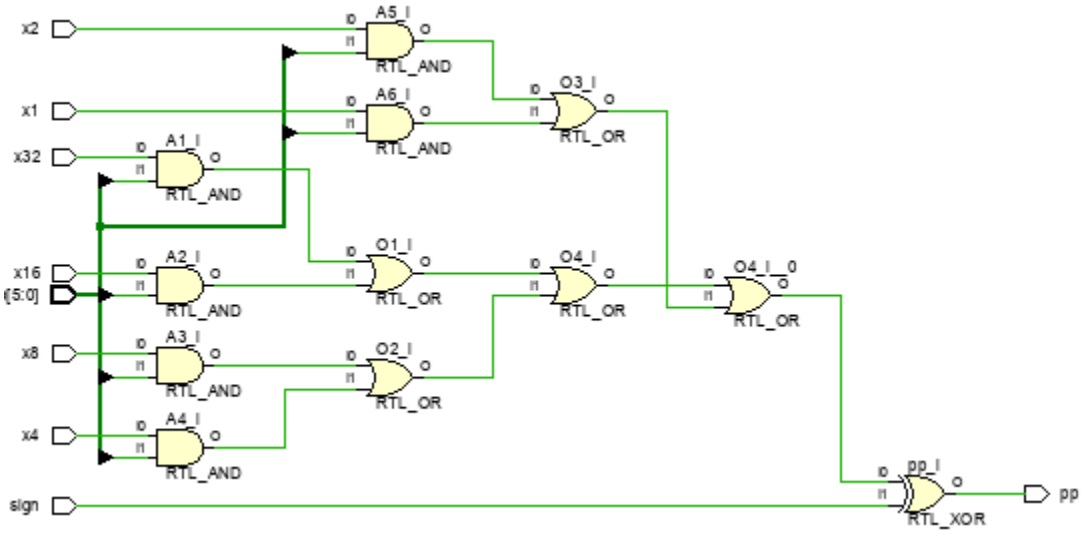


Figure .Radix-2^6Partial Product Generator

1.1. Rounding

In this method, the Least significant bits (LSB) of the partial products are discarded. In order to compensate for the loss of accuracy, a new correction term is introduced during accumulation of the result. Based on the number of bits that are discarded in each partial product, rounding can be classified as: Asymmetric Rounding and Symmetric Rounding.

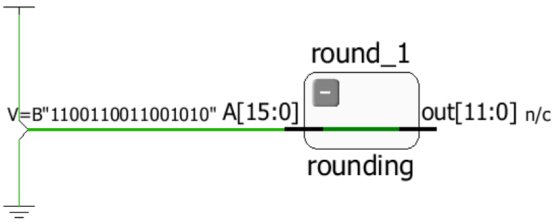


Figure .Rounding Block

In Asymmetric Rounding, the number of bits discarded is different for each partial product. More bits are discarded from the least significant partial products and less bits are discarded from the most significant partial products. In Symmetric Rounding, the number of bits discarded is the same in all the partial products. For the purposes of implementation, we have considered an Asymmetric Rounding Block.

1.2. Perforation

In this method, ‘p’ successive rows of partial products are eliminated completely. Generally the ‘p’ successive rows start from the least significant partial products. But any intermediate

rows can also be eliminated as per the requirements. For the purposes of implementation, we have considered perforation starting from the least significant partial product row.

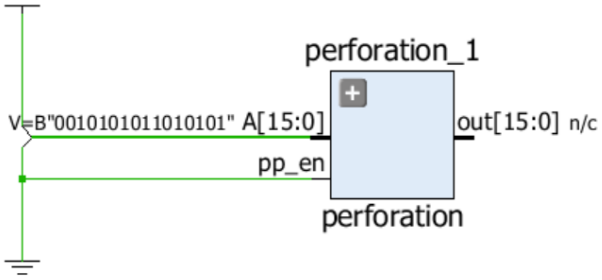
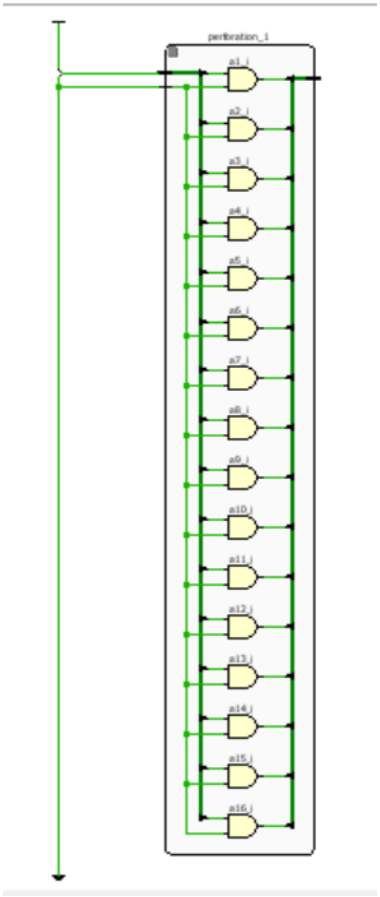


Figure .Perforation Block



Implemented Architectures

In the light of improving efficiency and unit gate savings without compromising the accuracy of the multiplier, novel approximate multiplier architectures have been proposed that is a coalescence of conventionally adopted inexact concepts discussed in the above section. Feasible combination of the architectures are

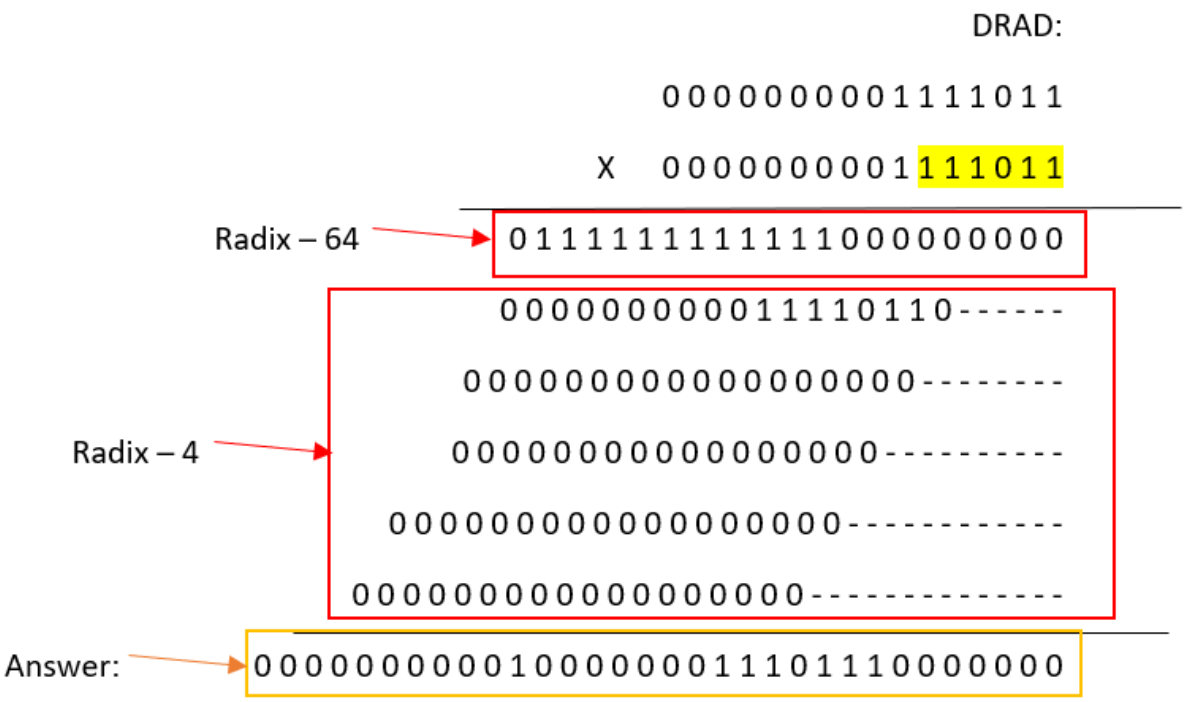
1. High Radix and High Radix
2. High Radix and High Radix with Rounding
3. High Radix with Rounding
4. High Radix with Rounding and Perforation
5. Rounding with Perforation

An expiation of the above proposed architectures are given below with simulation results.

1.3. High Radix & High Radix Multiplier

The architecture of this multiplier has been designed to work on two 16-bit inputs. The high radix logic is applied to the multiplier. The six bits in the LSB are applied to the radix-64 encoder and partial product generator. The remaining bits along with MSB of radix-64 encoded bits are operated by a radix-4 encoder and partial product generator, taking three bits at a time in overlapping fashion. The number of bits in partial product output from radix-64 is 22 bits and that of radix-4 is 18 bits. Radix-64 is made to generate one partial product and radix-4 produces 5 partial products. These partial sums inflict a shift of two on the succeeding leaf of partials if generated by radix-4 module and six if generated by radix-64 module.

The partial products after being shifted appropriately, are added using half adder and full adders effecting a 32-bit sum.

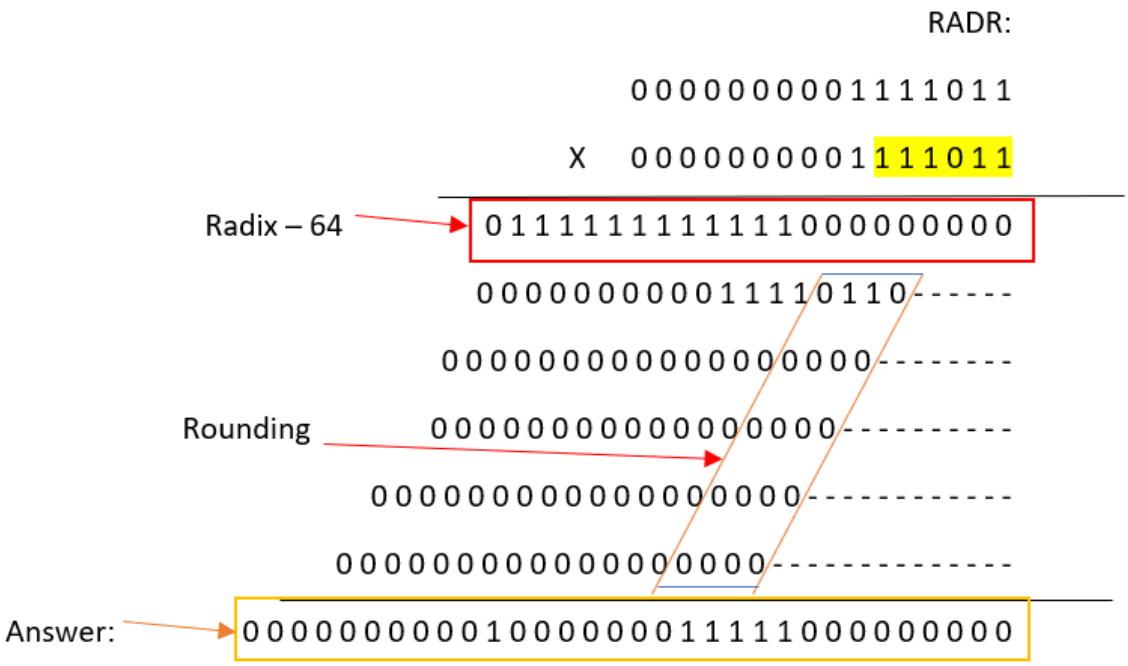


1.4. High Radix & High Radix with Rounding

This design of the multiplier encourages the use of high radix concept with rounding aiming at extensively reducing the area utilisation by the structural unit and energy efficiency by curtailing the bits generated.

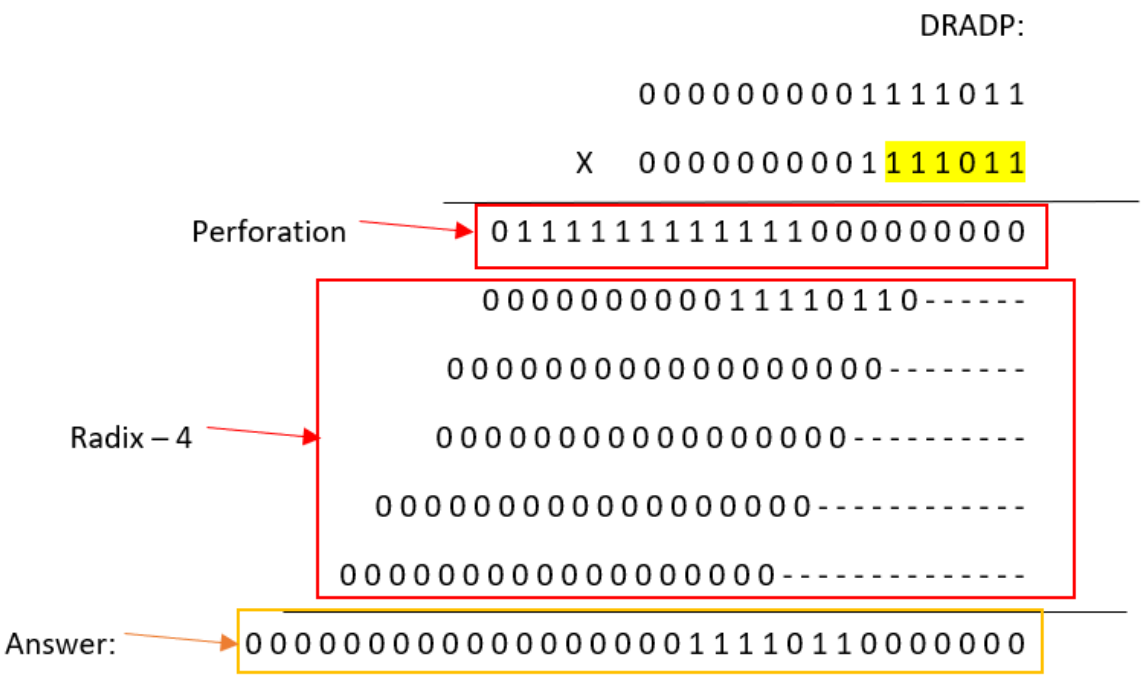
The multiplier accepts two 16-bit inputs, multiplier bits are raised to radix-4 and radix-64 in a split. Least six significant bits are encoded to radix-64 and the remaining bits are encoded to radix-4. A 22-bit partial product of radix-64 and five 18-bit partial sums of radix-4 are positioned for summation.

Full adder and half adder modules are cumulatively used rippling the sum and carry between them in decrypting a 32-bit output. (rounding details)



1.5. High Radix and High Radix with Perforation

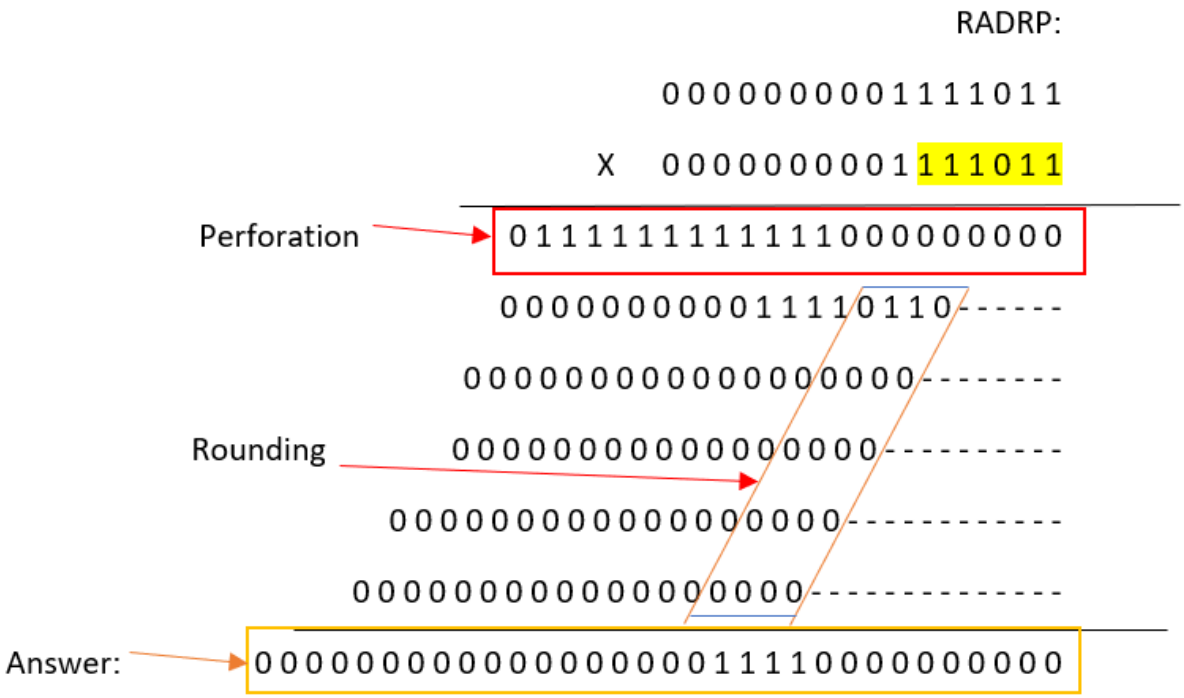
This design of the multiplier encourages the use of the high radix concept with perforation aiming to increase the approximations. Here, the LSB bits of partial products are eliminated by a constant factor. It delivers high error values,(almost 50%).



1.6. High Radix with Rounding & Perforation

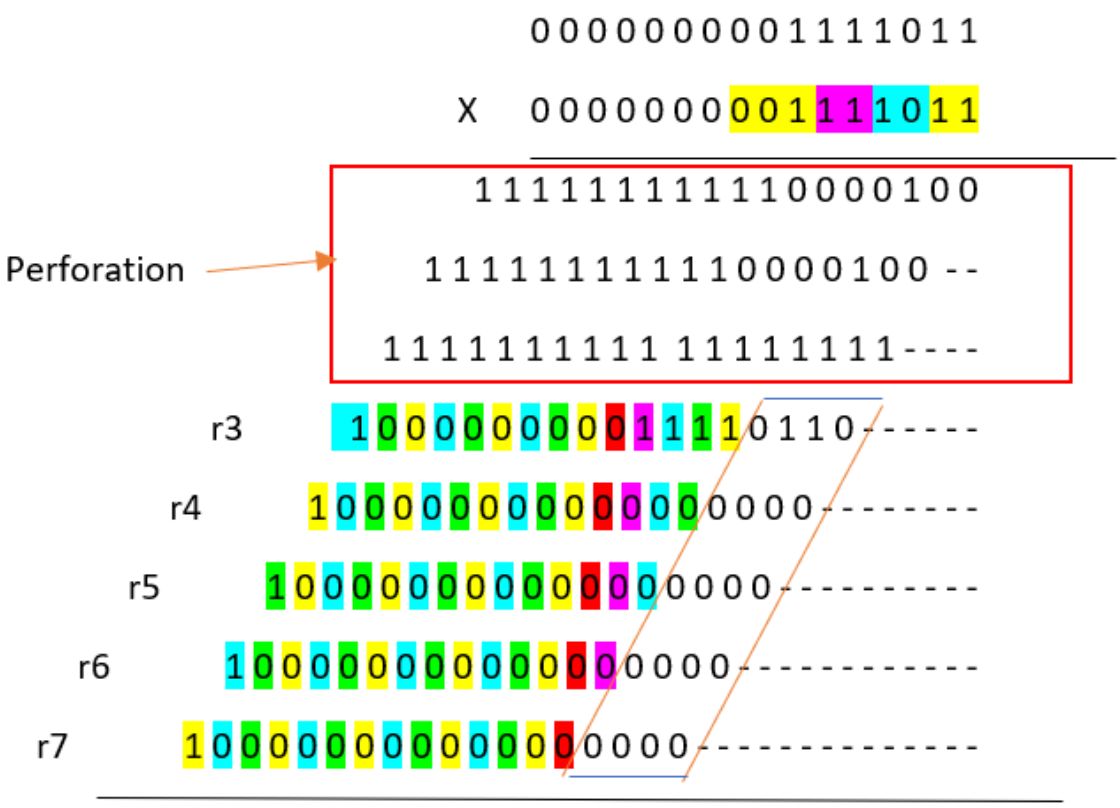
This pattern is equivalent to combining rounding with perforation, since removing the radix-2k partial product is equivalent to the perforation of k/2 radix-4 partial products.As a result of which, the approx configurations for RADRP are p for perforation and t for

rounding. This technique generates small error scaling, around 4 %.



1.7. Rounding with Perforation

This architecture of the multiplier benefits from the concept of Rounding and Perforation unlike radix-4 logic. The multiplier takes in two 16-bit inputs, a multiplier and a multiplicand. The partial products are generated using radix-4 encoder. Grouping a 16-bit multiplier into three, produces eight partial sums. However, since the perforation concept strikes out partial products generated by the least two significant bits, the logic is affected by not implementing radix-4 modules for the same. This accounts for seven 18-bit partial products whose least four significant bits are rounded and positioned for operation in ALU.


0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1
X 0 0 0 0 0 0 0 0 0 1 1 1 0 1 1
1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 0
Perforation
1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 1 0 0 --
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 ----
r3 1 0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 1 0 - - - - - -
r4 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - - - - - - -
r5 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - - - - - - - -
r6 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - - - - - - - - - -
r7 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 - - - - - - - - - - - -

2. Literature Survey

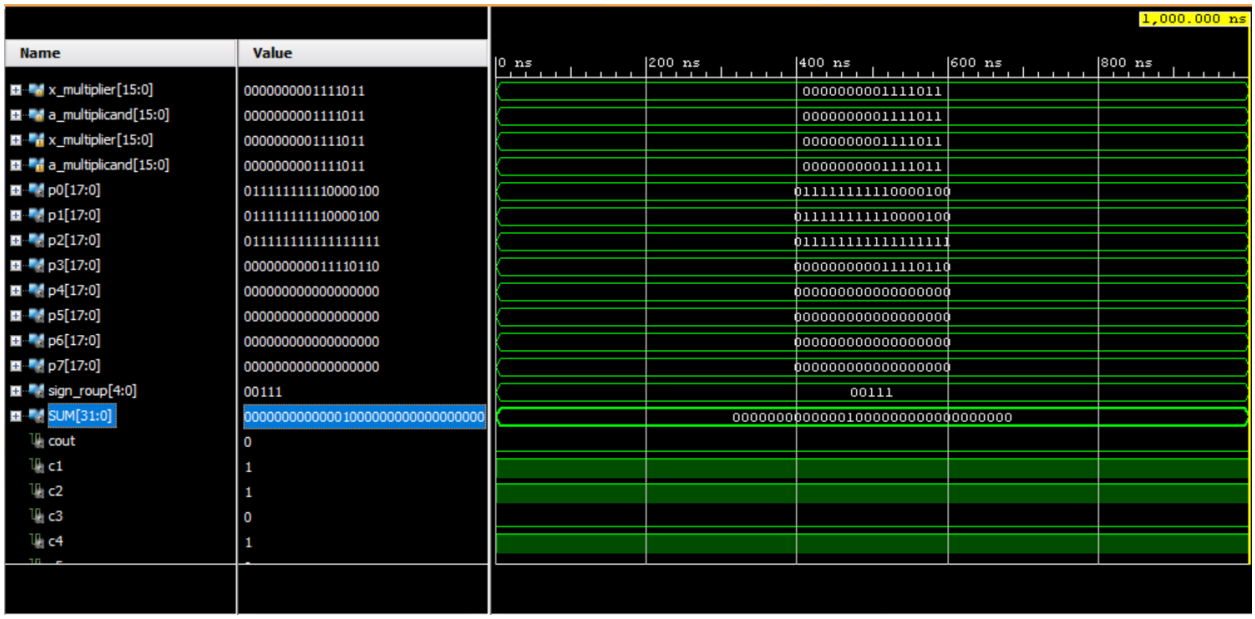
S.No	Paper, year	Author Name/s	Content	Limitation
1	Cooperative Arithmetic-Aware Approximation Techniques for Energy-Efficient Multiplier	Vasileios Leon, Konstantinos Asimakopoulos, Sotirios Xydis, Dimitrios Soudris, Kiamal Pekmestz	Multipliers using approximate computing, using 5 different hybrid techniques that leads to 60% energy saving	Very less accuracy

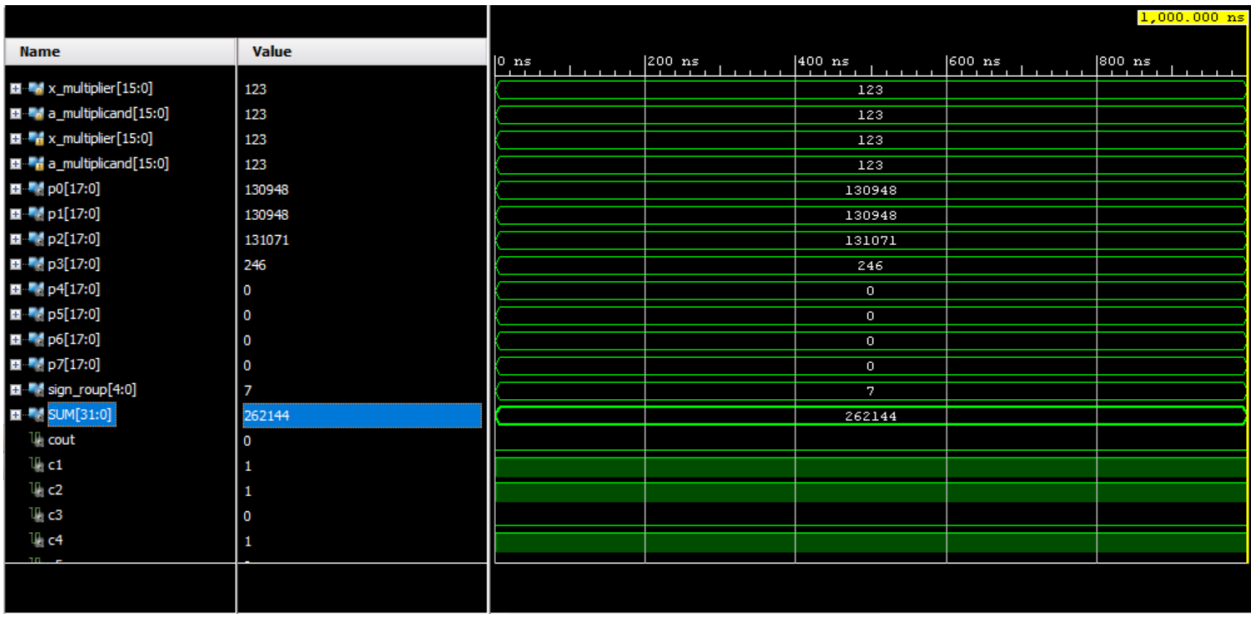
2	**Approximate Hybrid High Radix Encoding for Energy-Efficient Inexact Multipliers, March 2018**	**Vasileios Leon , Georgios Zervakis , Dimitrios Soudris, and Kiamal Pekmestzi**	**Circuits and equations for partial product generation**	**Accuracy is compromised to small extent , power saving upto 40%**
3	**Design-Efficient Approximate Multiplication Circuits Through Partial Product Perforation, October 2006**	**Georgios Zervakis, Kostas Tsoumanis, Student Member, IEEE, Sotirios Xydis, Dimitrios Soudris, and Kiamal Pekmestzi**	**Application of software based perforation technique in hardware**	**outperforms above processes in terms of power dissipation and error**

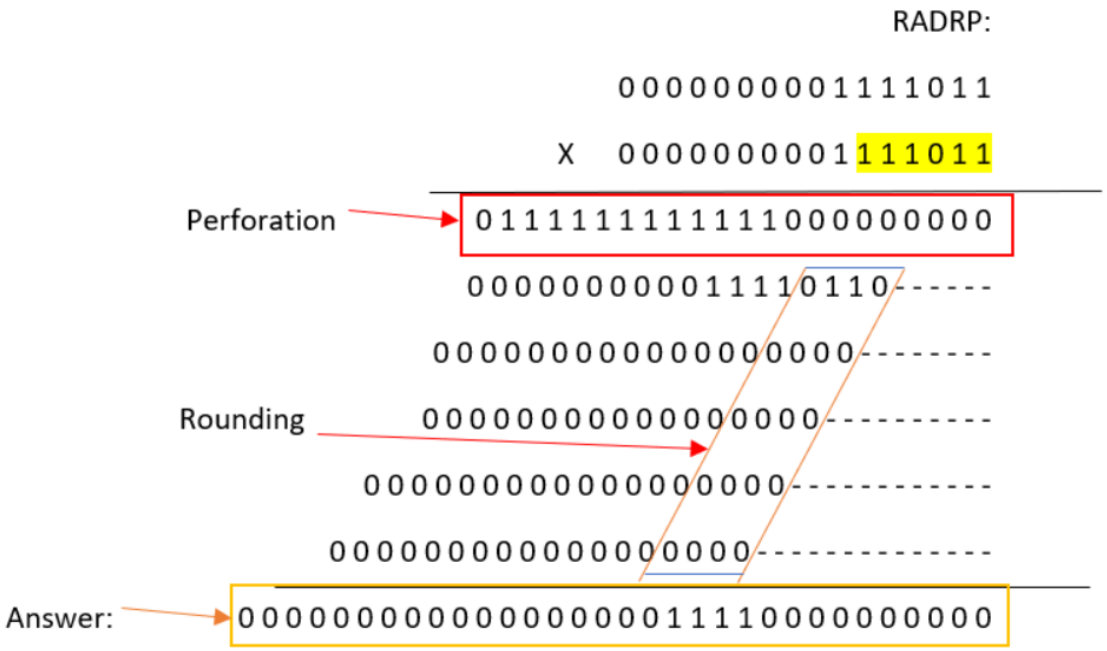
4	**Design of Radix-4,16,32 Approx Booth Multiplier Using Error Tolerant Application**	**Gunjan Jain1 , Meenal Jain2 , Gaurav Gupta3**	**Idea of circuit of a 4, 8, 16, 32 bit booth multiplier**	**Can be used for error tolerant systems only.**

Implementation and Results

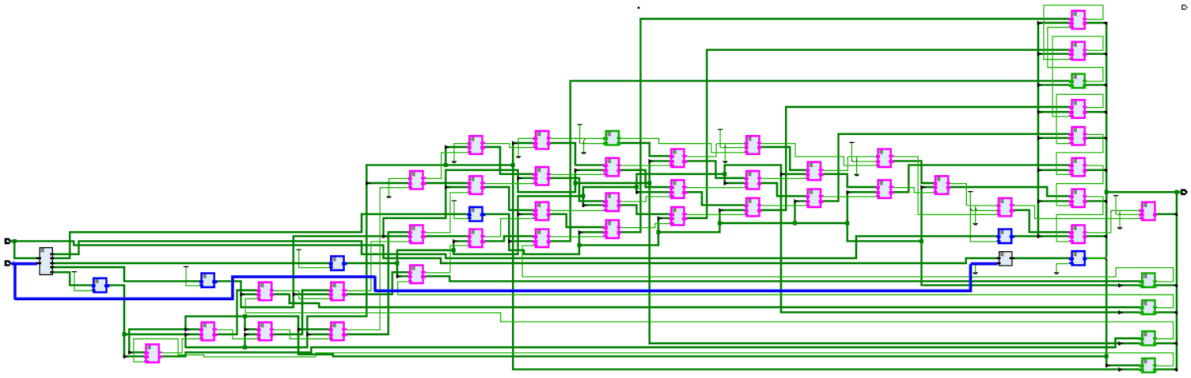
1)ROUP

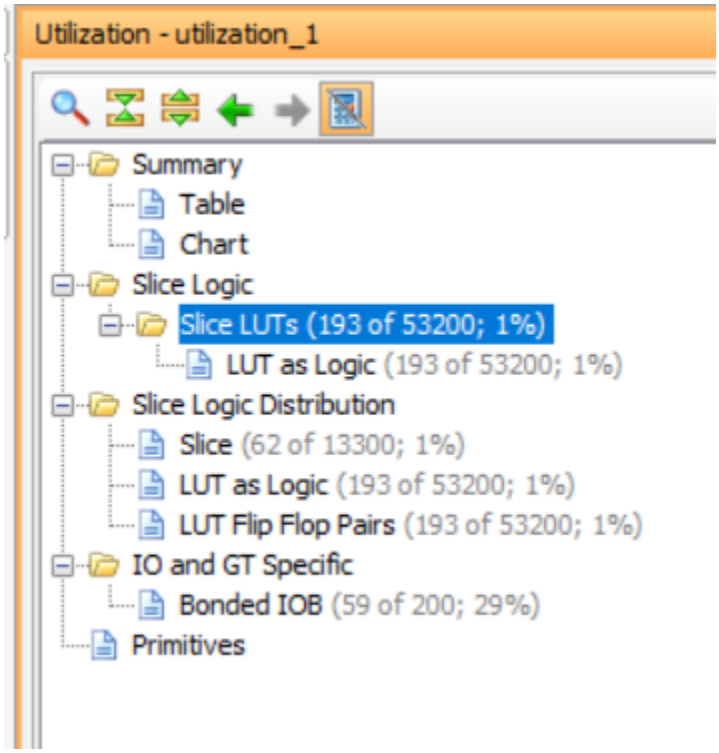




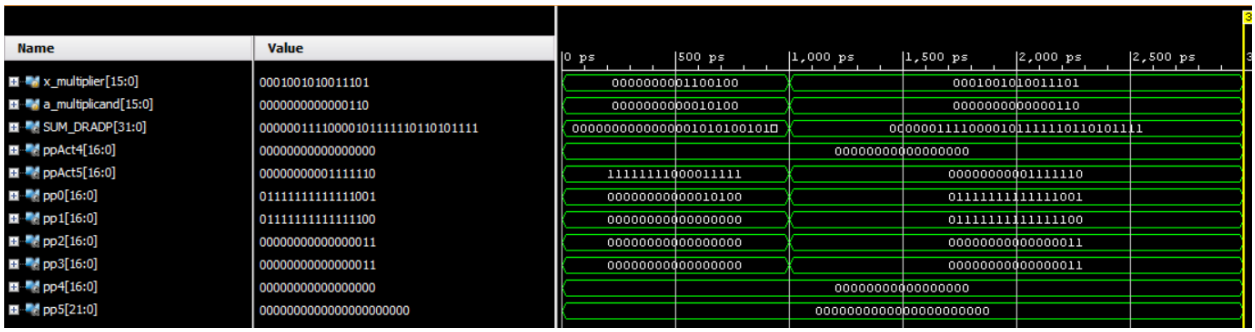


2)DRADP schematic generated:

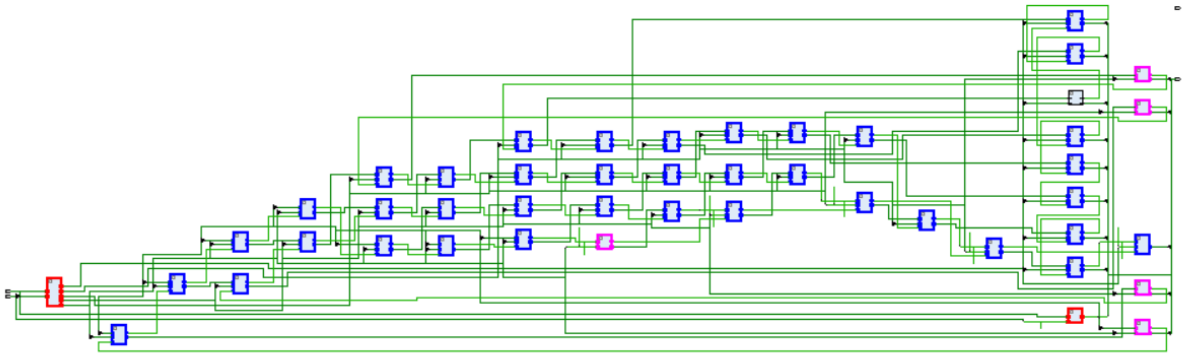


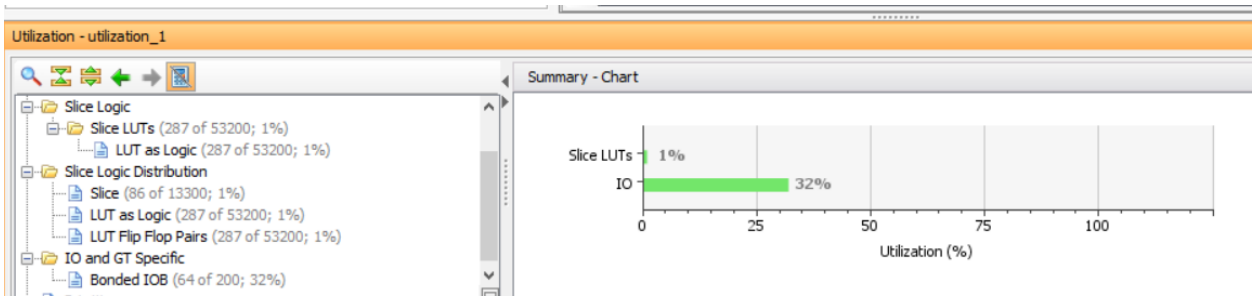


Simulation results

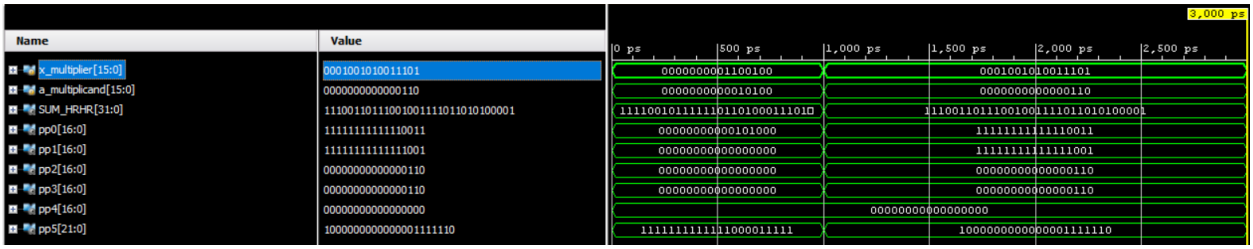


3)DRAD



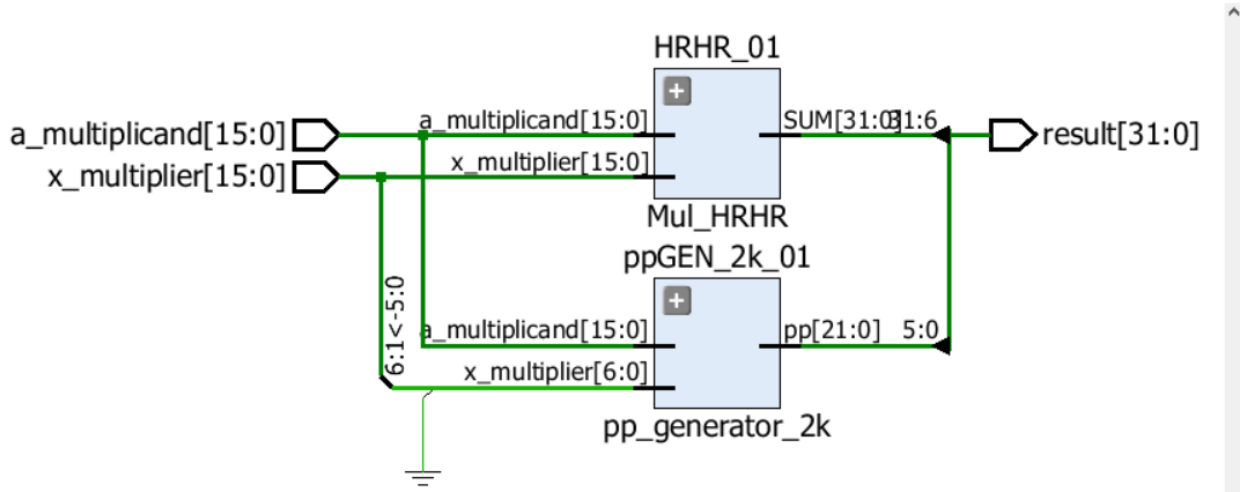


Simulation Results

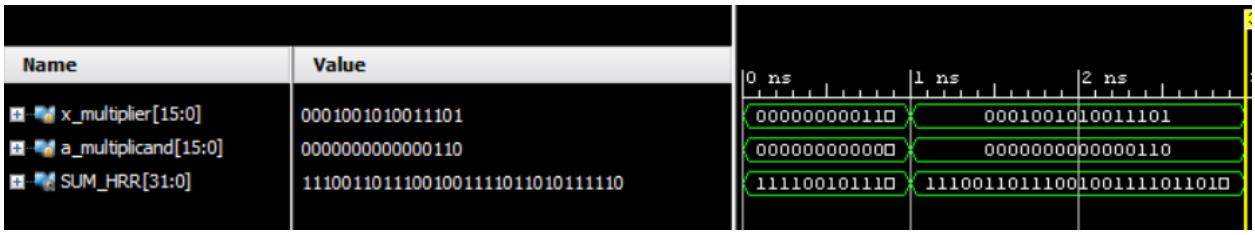


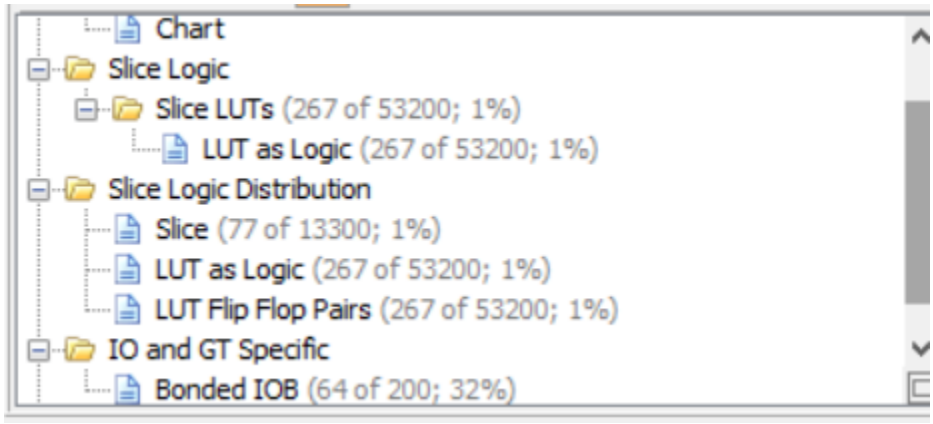
4)RADR

Schematic



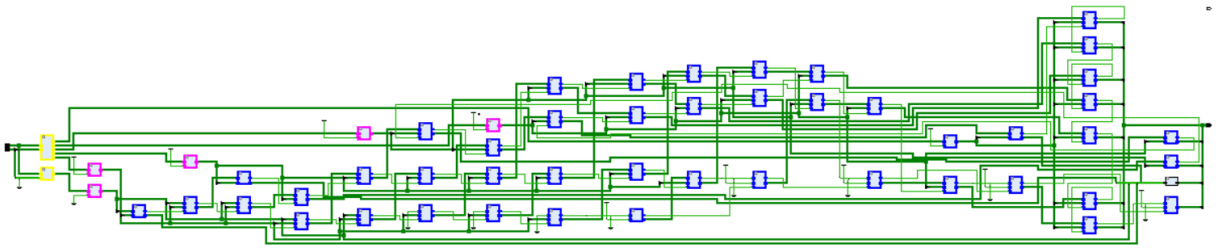
Simulation results



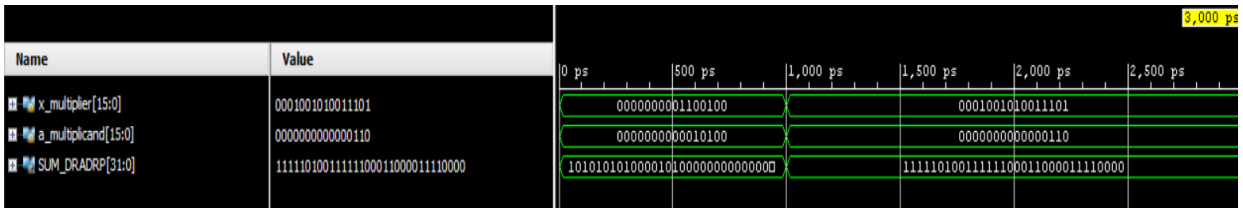


5)DRADRP

Schematic



Simulation results



Conclusion and Future scope

The implemented paper does an extensive exploration of using cooperative arithmetic-level approximation techniques to create inexact multipliers.To exploit fully the usage and application of approximation techniques the 5 hybrid multipliers has been tested against various features like size , error, power consumption, delay, and has been seen to outperform other multipliers in different aspects.

References

[1] V. Leon, G. Zervakis, D. Soudris, and K. Pekmestzi. 2018. Approximate Hybrid High Radix Encoding for Energy-Efficient Inexact Multipliers. IEEE Transactions on Very Large Scale Integration Systems 26, 3 (March 2018), 421–430

[2] C. Liu, J. Han, and F. Lombardi. 2014. A Low-Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery. In Design, Automation and Test in Europe. 1–4.

[3] Zhu, N., Goh, W. L., Zhang, W., Yeo, K. S., & Kong, Z. H. (2010). Design of low-power high-speed truncation-error tolerant adder and its application in digital signal processing. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 18(8), 1225-1229.

[4] G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, “Design-efficient approximate multiplication circuits through partial product perforation,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 24, no. 10, pp. 3105–3117, Oct. 2016.