International Journal of Recent Advances in Multidisciplinary Topics
Volume 2, Issue 12, December 2021
https://www.ijramt.com | ISSN (Online): 2582-7839

Design and Implementation of 32 Bit Systolic Array Matrix Dadda Multiplier

Suparna S. Nair[1*], C. H. Deepak[2]
[1]*Student, Department Electronics and Communications Engineering, VIT-AP University, Amaravathi, India*
[2]*Professor, Department Electronics and Communications Engineering, VIT-AP University, Amaravathi, India*

***Abstract*: The most principally used operation in mathematics is multiplication. The real world applies integer multiplication whereas binary multiplication is applied for integer multiplication. The efficient algorithm to perform a binary multiplication would be systolic algorithms. Systolic array algorithms reinstate a pipeline structure with an array of processing elements that can be implied to perform a prevailing operation. This paper is developed with the perspective of designing a 32-bit systolic matrix Dadda multiplier in Icarus v10.**

***Keywords*: Systolic array, 32 Bit, matrix Dadda multiplier.**

1. Introduction

Systolic array multiplier is a processor arrangement in array Flow of data is synchronous across the array between neighbors, normally for contrast data glide with their contrast directions. Systolic array multiplier is an effectual multiplier that conceptualizes pipelining in magnifying the speed of multiplication. Systolic array algorithms are a form of pipelining which it's more than one dimension. The data stream from a memory passes through many processing elements and returns to memory. A systolic array is a hardware structure built for fast and efficient operation of regular algorithm they perform the same task with different data at different time prompt. Systolic architecture offers competence to uphold the high throughput capacity requirement. A systolic array has the characteristic feature of modularity, regularity, local interconnection, a high degree of pipelining, and highly synchronized multi-processing. Since many cells drawn from a small set of cell types with bounded degrees relate to neighboring cells in a regular fashion, systolic arrays are easily realized in contemporary VLSI technology. This work manifests an effectual design and methodical implementation of the 32-bit systolic array matrix Dadda multiplier. Dadda multiplier is a binary multiplier that uses the concept of column compression to reduce the partial products and hence increases the speed of multiplication.

2. Methodology

In systolic multiplication, to carry out the multiplication and get the final product following steps should be followed:

- The multiplicand and multiplier are arranged in the form of an array
- Each bit of the multiplicand is multiplied with each bit of the multiplier to get the partial products.
- The partial products of the same column are added along with the carry generated.
- The resulted output by adding partial products and carry is the final product of the two-binary number.

1) Types of Methodology

- Array multiplier
- Wallace multiplier
- Dadda multiplier

2) Dadda multiplier

Dadda Method does minimalistic reduction required at each level, same number of levels as Wallace multiplier. In the dadda multiplier, fewer reductions are carried out in earlier stages. The dadda multiplier architecture can be divided into three stages:

- Generation of partial products
- Partial product reduction
- Matrix reduced to, remaining rows are added using adder.

3. Proposed Architecture

The Dadda multiplier is a hardware multiplier design that is slightly faster and requires lesser gates when compared to the array multiplier.

Dadda multiplier has 3 steps:

- Multiply each bit of the argument, yielding an N2 result
- Two-layer partial product reduction (i.e. full adder and half adder)
- Group the wire in two numbers and add them with the adder.

1) Implementation of the Multiplier

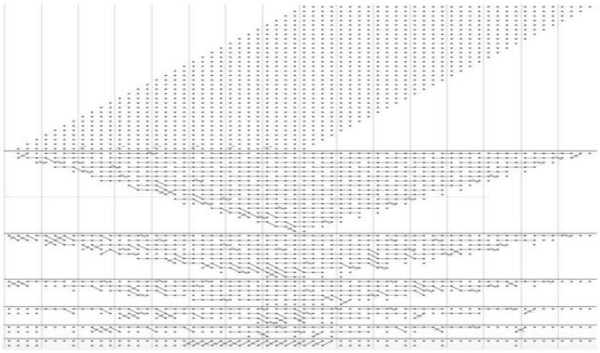
The multiplication of M bit multiplication by N bit multiplier yields an N by M matrix of partial products. Each full adder will be accepting three inputs whereas half adder will be accepting inputs from their given column and would be producing a sum and carry, where they carry it would be going into the next more

*Corresponding author: suparnanair14@gmail.com

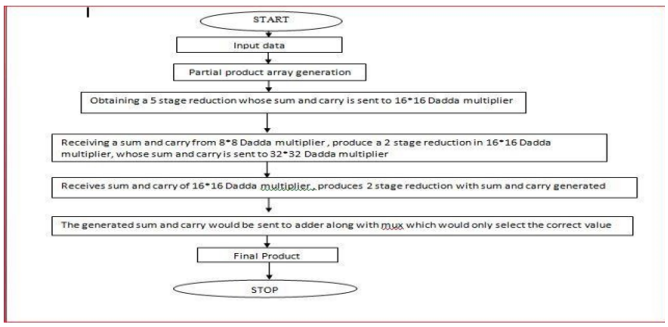
significant column. The implemented 32*32 dadda multiplier with the dot diagram is given below. The data scheme minimizes the number of adder stages required to perform the summation of partial products. This is achieved by using full adder and half adder to reduce the number of rows in the matrix of bit at each summation stage by a factor of 3/2.

The entire 32*32 Multiplication requires eight stages. Always the first stages partial product stage, which is obtained by simple multiplication of multiplicand with the multiplier. The number of rows present at the stage is 32. Now reduce the number of rows further in such a way that the final stage contains only two rows. For which dadda introduces a sequence of intermediate matrix height that provides the minimum number of reduction stages for a given size multiplier. This sequence is determined by working back from the final two rows matrix, limiting the height of each intermediate matrix to the largest integer that is no more than 1.5 times the height of its successor.

In the dadda multiplier implementation, in general, the number of full adders required is N2-4N+3 whereas the number of half adders would be N-1. The single bit in the first column of the first stage represents the least significant bit of the product. From the dot diagram lower row stage is derived from their higher row stage (i.e.,) 3-row stage would be derived from the 4-row stage, 2-row stage from a 3-row stage with the help of (3, 2) and (2, 2) counters. The data obtained would be flowed to carry select adder in the final stages which sum the correct value of the partial product and gives back the final value.



2) Flow Chart



4. Result

Fig. 1. Shows the obtained result of the design incorporated using Verilog HDL. The design and methodology have been successfully incorporated.

5. Conclusion

Design and implementation of 32bit systolic array of with two dimensional inputs and output by implying a Dadda multiplier whose result is summed and displayed, its realization is carried out in Icarus Verilog.

Acknowledgment

This project was supported by the faculty, director, and vice-chancellor of VIT-AP. I would like to thank my guide for providing immense support and encouragement, without which this paper would've been impossible.

References

[1] S.Subathradevi, C Vennila, "Systolic Array Multiplier for Augmented Data Centres Network Communication Link", Springer, 24 February 2018.

[2] Pritham H Langade, S.B Patil "Design of Improved Systolic Array Multiplier and Its Implementation on FPGA", International Journal of Engineering Research and General Science vol.3, no. 6, November-December 2015

[3] Khumanthem Devjith Singh, K. Jyothi “Design and Implementation of VLSI 8 bit Systolic array multiplier", International Journal of Advanced Research in Electrical, Electronics, and Instrumentation Engineering(IJAREEIE) vol.3, no. 11 November 2014.

[4] H.T.Kung, “Why systolic architectures?” IEEECom., vol.15

[5] Naveen M P, Ebenezer "Simulation Result Analysis of Efficient Design Systolic Architecture", International Journal of Engineering Research and Technology(IJERT),

[6] Rakesh Birle, Lalit Bandil," Design and FPGA Implementation of Systolic Array Architecture for Matrix Multiplication ", International Research of Engineering and Advanced Technology (IJEAT), vol.1, no. 6, August 2012.

[7] Pritham H Langade, S.B Patil "Design of Improved Systolic Array Multiplier and Its Implementation on FPGA", International Journal of Engineering Research and General Science vol.3, no. 6, November-December 2015

[8] Ganapathi Hegde, Cyril Prasanna Raj P, P.R Vaya, “Implementation of Systolic Array Architecture for Full Search Block Matching Algorithm on FPGA, European Journal of Scientific Research, vol-33, no-3,2009.

[9] H.T.Kung,” Why systolic architecture?”IEEE Com.,vol.15