Modifying a Multiply Accumulate Unit(MAC) design, which uses Radix-2Booth encoding with Vedic Multiplier

Manmohan Lonawat, *Jin H. Park*

Abstract—The digital signal processing system(DSP) is designed for taking real-world signals like audio, video, position, etc., and to have them digitized, which can be manipulated using mathematics. So, various performance applications use the specially designed Multiply-Accumulate (MAC) unit, which itself is a major part of the DSP, for filtering, coefficient multiplication, and many other applications. We are going to use the existing MAC unit model, which makes use of the Radix-2 booth multiplier. In this work, we plan to replace the design by changing the multiplier with a Vedic multiplier and then design a 32-bit MAC using the HCA(half carry adder). Now, this modified Multiply Accumulate Unit(MAC) design with Vedic multiplier is being used to improve the performance on the aspect of delay. It is supposed to result in higher performance than the original. This design will then be synthesized and simulated using Cadence Innovous and ISE Xilinx. Now we compare and analyze the two different MAC unit designs. The proposed work is supposed to provide much better performance as compared to the conventional pipelined carry propagate MAC units.

Index Terms—Digital signal processing system, DSP, MAC Unit, Vedic Multiplier, Half carry adder.

1 Introduction

1.1 Background

The current trend in VLSI design is to create low-power, low-area devices that are fast and efficient, due to which we must reduce the latency in our designs. In modern digital signal processing(DSP), the multiply-accumulate operation is the most common, but also the most important part. The MAC unit's heart is a Digital Multiplier. Multiply and add functions are performed by the MAC unit. It has two stages of operation. It first calculates the product of two numbers and then passes the result on to the second stage operation, addition/accumulation.

1.1.1 *Vedic Multiplier*

Vedic Mathematics is an ancient Indian mathematical discipline that may be directly applied to other branches of mathematics, including arithmetic, algebra, and so on. Vedic Multiplier boosts multiplier speed by using Indian Vedic sutras. This archaic style of Vedic Maths is based on the sixteen Vedic Sutras, which define the natural approach to solving mathematical problems. It simplifies the process of computing any outcomes. It has the ability to solve arithematic-related difficulties quickly and easily. It contains 16 sutras that can be applied to problems in geometry, calculus, conics, algebra, and arithmetic.

1.1.2 *Booth's Multiplier*

Booth multiplier follows Booth's multiplication algorithm invented by Andrew Donald Booth in 1950. It multiplies two signed binary numbers in two's complement notation while preserving the sign of the result. It outperforms earlier methods of multiplication by reducing the number of iteration steps. It skims the multiplier operand and skips chains of the algorithm thus reducing the number of additions required to produce the result.

- *Manmohan Lonawat with the Department of Computer Science, California State University, Fresno .*
E-mail:manmohanml@mail.fresnostate.edu
- *J. Park is a faculty and Graduate Coordinator in the Department of Computer Science, California State University, Fresno*
E-mail: jpark@mail.fresnostate.edu

Manuscript received December 04, 2021

1.2 Problem Statement/Motivation and Potential Impact

There are many operations, which are of high performance that are needed to perform the following operations- convolution, digital filtering, and the fast Fourier transform(FFT). In general, the power consumption by a DSP (digital signal processing) unit is entirely dependent on the MAC unit. The goal of this project is to create a MAC that may be used to develop effective signal processing algorithms.

1.3 Proposed Method

At the accumulation stage, it performs multiplication between input operands and then adds the result to the previously produced multiplied result. In a digital system, the multiplier necessitates a significant amount of delay, hence it determines the critical path. MAC unit's performance is significantly based on the multiplier. In this work, we use the Vedic Multiplier, which supposedly offers a systematic design for a quick and efficient MAC unit.

1.4 Hypotheses

Adders are also an important component of any computer system. Adders are an important component of the MAC unit. As a result, choosing a multiplier and adder with a short latency and low power is critical. We perform the work in two different sections. In the following section, we present the design technique for an NxN bit Vedic

Multiplier, and in the later sections, we further provide the design of a Multiply Accumulate Unit, which employes the Vedic Multipliers.

2 RELATED WORK

2.1 Method

A survey is conducted in this work for several types of MAC units with various multipliers and adders, where multipliers are used to make partial products and adders are used to collect these partial products. This research looks at a variety of MAC units that have been created with fast speed and low power consumption in the past [1]. In order to optimize the functioning of a MAC unit, a survey or overview of the many devices employed in it is conducted. [2]. This work uses Vedic Mathematics to offer a systematic design for a fast and efficient MAC unit. This is accomplished by describing the design technique for a NxN bit Vedic Multiplier, followed by an explanation of how to create a Multiply Accumulate Unit using Vedic Multipliers [3]. The general technique for NxN multiplication is proposed in this study. This reduces the time it takes to calculate the NxN bit's multiplication result [4]. To increase the MAC unit's delay performance, a modified Radix-4 booth multiplier method is implemented. The counters (6,3) are utilized for partial product reduction and accumulate operations. As a result, the system runs at a high level of efficiency [5]. The efficiency of a novel multiplier-and-accumulator (MAC) architecture for high-speed arithmetic is increased by coupling multiplication with accumulation and designing a carry-lookahead adder (CLA). A high-speed adder is used to construct the Modified Booth multiplication algorithm. Multiplication is sped up with the use of a high-speed adder [6]. Urdhva Tiryagbhyam is one of the multiplication methods employed in Vedic mathematics, and it is a 32 X 32 bit multiplication. Urdhva Tiryagbhyam is a multiplication formula that can be utilized in any circumstance. The adder that was used was Carry Look Ahead. The proposed design outperforms the carry-save adder in terms of performance [7]. In this work, the performance metrics of adders such as area and latency are determined and compared for Ripple Carry Adder (RCA), Carry Skip Adder (CSA), Carry Increment Adder (CIA), Carry Look Ahead Adder (CLA), Carry Save Adder (CSA), Carry Select Adder (CSA), and Carry Bypass Adder (CBA). Verilog HDL is used to create a variety of adders [8]. The Vedic Multiplier for standard CMOS and Complementary Pass TransistorLogic has been built and examined in this paper (CPL). Vedic Multiplier is a 4-bit and 8-bit multiplier that uses ordinary CMOS and CPL gates. Their speed, area, and power are all compared and assessed [9]. In this study, we propose a 32-bit MAC based on a 32-bit array multiplier and RCA (ripple carry adder), as well as a MAC based on Vedic multipliers and RCA. Another modified MAC unit in the paper employs a Vedic multiplier and carry-save adder [10].

2.2 Novelty

Adders are a critical component of any computing system in Digital Signal Processing (DSP) applications as in DSPs speed is crucial. Adders are an important component of the MAC unit. As a result, selecting an adder with a short latency, low power, and high speed is critical [1]. The MAC unit determines the processor's overall power consumption and computational latency. As a result, it's critical when building a MAC unit that uses less power, has less delay, and takes up less space. In the design of real-time signal processing and DSP systems, this is a fundamental concern [2]. High-performance multiply accumulate operations are required for operations like digital filtering, convolution, and the fast Fourier transform (FFT). A Multiply Accumulate Unit has been built using this multiplier, which exhibits shorter computation time when executing MAC operations [3]. Many digital signal processing operations necessitate multiple multiplication, which necessitates the use of an extremely fast multiplier to meet a wide range of hardware and speed requirements. Based on Vedic mathematics, this work proposes a systematic design process for a fast and area efficient digit multiplier [4]. Because low power multipliers are commonly used in DSP applications, they must be designed. As a result, the goal of this project is to create a MAC that may be used to develop efficient signal processing algorithms [5]. When compared to traditional approaches, the Booths multiplier minimizes the amount of iteration steps. A Booth Encoding 8 bits Multiplier is built in this work because it is an excellent approach for considerably increasing algebraic speed. It is also used to reduce the amount of partial products formed in a multiplication process using the modified Booth algorithm [6]. A new algorithm based on Vedic mathematics is being developed. The usage of Vedic mathematics can help to simplify and even optimize traditional mathematical algorithms [7]. The fundamental building blocks of digital systems are arithmetic units. For the efficient implementation of arithmetic units, adders have become a critical hardware unit. The goal of this study is to use the Verilog language and Xilinix ISE to develop and simulate several types of adders. After that, the various adders' performance parameters are calculated and compared [8]. In terms of area, delay, power, and complexity, the Vedic multiplier outperforms the Booth Wallace tree multiplier. In this study, the various applications of the Vedic Multiplier are examined [9]. The recent trend in VLSI design is to create low-power, low-area devices that are also quicker in nature, thus it is needed to reduce the latency in existing designs as we, which is one of the work done in this paper [10].

2.3 Results/Findings/Analysis

Comparing the different Adder topologies, it was found that RCA has the best results in terms of area and power, while CSA is the worst but better in terms of delay, while in comparison of the multipliers it was found that overall Array multiplier has the best results in terms of complexity, area, cost than others, while the Dadda Multiplier is the worst [1]. In this survey it was found that some designs concentrated on increasing adder performance, while others focused on increasing multiplier performance, aslo in several architectures both the adder stage and the multiplier circuit have been altered [2]. In this paper, the results of device utilization, power dissipation and combinational delay is generated, while also generating simulation result for 16 bit MAC unit which has been designed. It shows far less computation time for performing MAC operations. Finally, the design complexity for large inputs is reduced in this study, while modularity increases, allowing it to be employed efficiently in a variety of Digital Signal Processing applications [?]. The results in this research show that for inputs with a large number of bits, design complexity decreases while modularity increases [4]. Depending on which n-bit MAC units are used, the performance analysis varies significantly. The proposed structure allows for quick multiply-add operations without the use of a pipeline [5]. Radix-2,4 Booth Multipliers are used here, and the entire procedure is designed to increase the speed of operation. When speed and circuit complexity are compared, the Radix-4 Booth Multiplier has a faster speed and a lower circuit complexity than the Radix-2 Booth Multiplier [6]. In this article, the computation latency for a 16x16 Vedic multiplier is 21.4ns, with an 8 percent device utilization, indicating a gain in performance [7]. In comparison to previous adder topologies, the results presented in this work imply that the Carry Increment Adder provides greater performance in terms of area and delay [8]. The simulation results for a Vedic Multiplier in this research show that CMOS logic outperforms CPL transistor logic. When compared to CPL, CMOS has less delay, lower power dissipation, and fewer transistors, hence it is superior [9]. In this paper, it was discovered that a MAC unit using a vedic multiplier and RCA was 33 percent more efficient in terms of delay than conventional MAC units. In terms of space, the MAC unit created using array muliplier and RCA was 37 percent more efficient than other MAC units. [10].

3 OUR APPROACH

3.1 Vedic Multiplier

As a building component, we'll need a 2X2 multiplier to create an 8X8 multiplier. Let's imagine we have two inputs, A (a1,a0) and B(b1,b0), each two bits wide, and the outcome is S (q3,q2,q1,q0), which is four bits wide. Due to vertical and crosswise multiplication, four partial products are formed during this multiplication: a0b0, a0b1, a1b0, and a1b1. The partial product a0b0 is directly transferred to the product S as q0, but the LSB of the sum of a0b1 and a1b0 is transferred to the product S as q1. After adding the carry bit to the partial product a1b1, q3 and q2 are determined to be MSB and LSB, respectively. Figure 1 depicts 2 bit multiplication using this method.

4 and 8 bit multipliers can be built with minor modifications using the similar approach as the 2x2 bit vedic multiplier. To put it another way, the input bits must be split into two halves of the input bit stream. Input bit streams (of say N bits) can be divided into (N/2 = n) bit lengths, which are further separated into n/2 bit streams, and so on until bit streams of width 2 are reached, which can be multiplied in parallel utilizing 2x2 multiplier blocks. Let's look at an example of a 4x4 multiplier, which is made up of four 2x2 multipliers. Inputs A and B are considered to be 1101 and 1010, respectively, and are divided into 2 bit chunks and multiplied in parallel using 2x2 multipliers. In the figure 2, there is an indication on how the 2x2 bits are multiplied.

The partial products q0,q1,q2,q3 are generated by 2x2 multipliers and are 4 bits wide. Similarly, as shown in Figure 3, these are added using three half adders to produce the final product S (q7,q6,q5,q4,q3,q2,q1,q0).

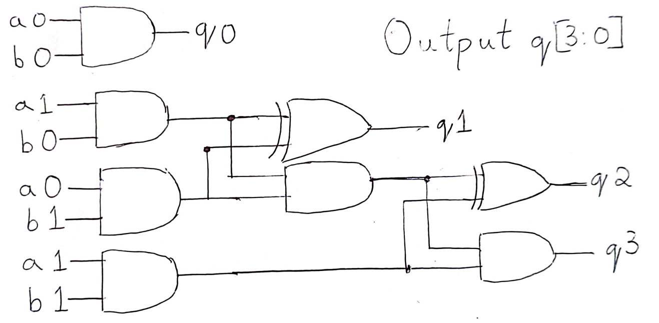


Figure 1: Block diagram of 2 bit Multiplier.

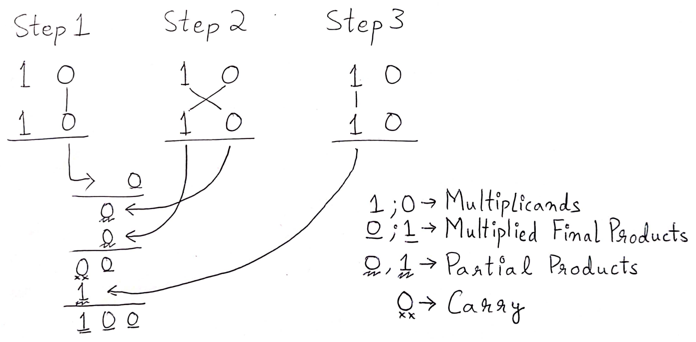


Figure 2: Multiplication of two 2-bit numbers.

Utilizing Adder 1, the sum of q2 (0010) and q1(0110) is

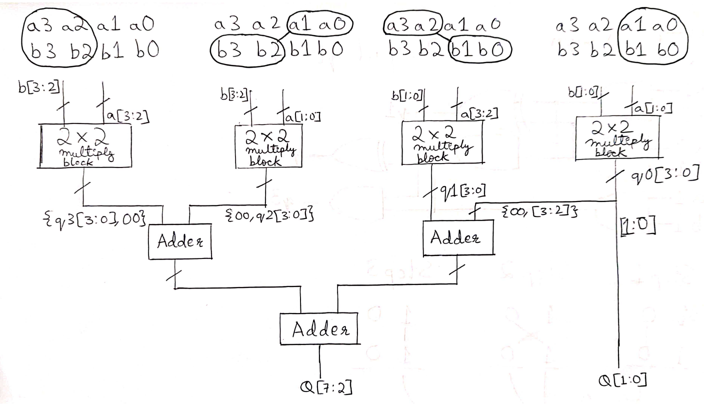


Figure 3: Design of 4x4 bit multiplier using 4 2x2 multipliers and 3 adders.

1000, utilising 0 as the carry bit. For addition, pipelined carry look ahead adders should be employed. The result, 1000, is added to the bit stream generated by concatenating the two most significant bits of q0, 00, and the two least significant bits of q3, 10, resulting in 1000. Adder 2 is used to combine them, yielding 0000 as the sum and 1 as the carry. The two least significant bits of q0 are directly passed

on to generate the end product's two least significant bits, S1,S0. S5,S4,S3,S2 is formed by the sum generated by Adder 2 i.e. 0000. A half adder is used to combine the carry bits generated by Adder 1 and Adder 2 to form a carry bit and a sum bit, which are 0 and 1 respectively. These are the MSB and LSB of one of Adder 3's inputs, the other being two MSBs of q3. S7, S6 are the outputs of Adder 3.
A 8 bit Multiplier unit has been created using the aforementioned approach of Vedic Multiplication, which is employed in the 8 Bit Multipliy-Accumulate Unit. Four 4x4 multiplier bricks are used to create the 8x8 multiplier. The multiplicands in this case are of bit size (n=8), but the result is of 16 bit size. For both inputs, a and b, the input is broken up into n/2 = 8 smaller chunks. These freshly produced 4-bit portions are provided to the 4x4 multiplier block, which then breaks them down even further into smaller pieces of size n/2 = 2 and feeds them to the 2x2 multiply block, precisely like the 4x4 Multiply block. The output of an 4x4 bit multiply block is 8 bits, thus the result is added using an addition tree with three full adders and a half adder. Extra zeroes must be padded at the MSB of one of the inputs for adder 3 in the case of 4 bit and 8 bit multipliers. Two more zeroes will be padded for an 8 bit multiplier. Figure 4 depicts the addition tree of a 8 bit multiplier.

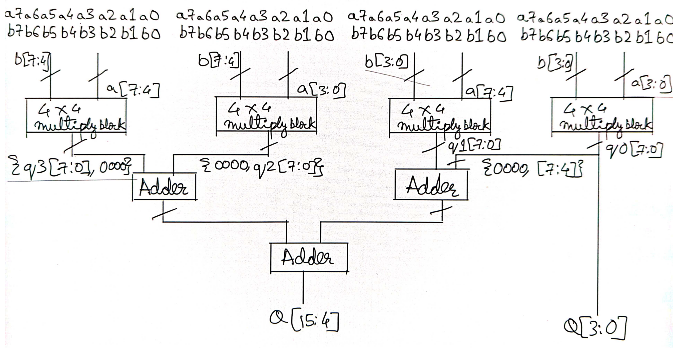


Figure 4: Design of 8x8 bit multiplier using 4 4x4 multipliers and 3 adders.

3.2 Booth's Multiplier

Booth's algorithm entails the encoding of multiplier bits as well as the production of partial products. Depending as to how many bits are utilized to encode the multiplier, different modified booths techniques have been developed. Modified booth algorithms, such as Radix 2 modified booth algorithms, lower the amount of partial products. Although NxN bit multiplication yields N partial products, modified Radix 2r yields N/r partials.

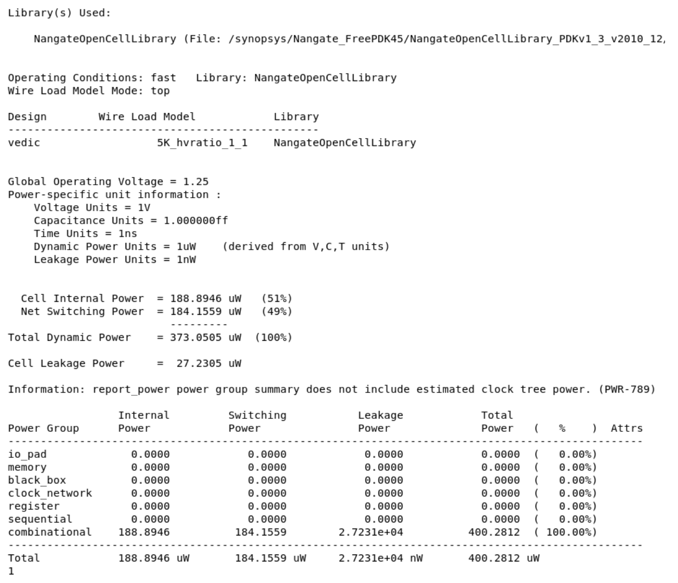
4 EXPERIMENT

4.1 Results

We have designed MAC unit with Vedic multiplier we have got the followiung Synthesis and Simulated Results using Cadence Innovous and ISE Xilinx respectively.

4.1.1 Synthesis Results

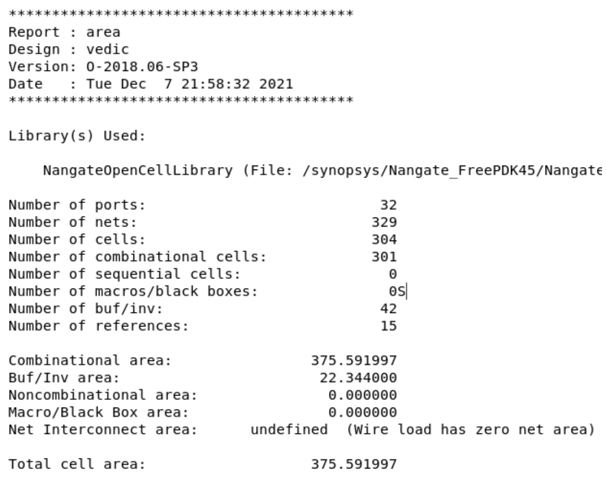
After we have designed MAC unit with Vedic multiplier we have got the followiung results using Cadence Innovous. Power required for the design of Vedic Multiplier is displayed in Figure 5. Area required for the design is displayed



```
Library(s) Used:

    NangateOpenCellLibrary (File: /synopsys/Nangate_FreePDK45/NangateOpenCellLibrary_PDKv1_3_v2010_12,


Operating Conditions: fast   Library: NangateOpenCellLibrary
Wire Load Model Mode: top

Design        Wire Load Model            Library
------------------------------------------------
vedic                  5K_hvratio_1_1    NangateOpenCellLibrary


Global Operating Voltage = 1.25
Power-specific unit information :
    Voltage Units = 1V
    Capacitance Units = 1.000000ff
    Time Units = 1ns
    Dynamic Power Units = 1uW    (derived from V,C,T units)
    Leakage Power Units = 1nW


  Cell Internal Power  = 188.8946 uW   (51%)
  Net Switching Power  = 184.1559 uW   (49%)
                         ---------
Total Dynamic Power    = 373.0505 uW  (100%)

Cell Leakage Power     =  27.2305 uW

Information: report_power power group summary does not include estimated clock tree power. (PWR-789)

                 Internal          Switching           Leakage            Total
Power Group      Power             Power               Power              Power   (   %    )  Attrs
--------------------------------------------------------------------------------------------------
io_pad             0.0000            0.0000            0.0000            0.0000  (   0.00%)
memory             0.0000            0.0000            0.0000            0.0000  (   0.00%)
black_box          0.0000            0.0000            0.0000            0.0000  (   0.00%)
clock_network      0.0000            0.0000            0.0000            0.0000  (   0.00%)
register           0.0000            0.0000            0.0000            0.0000  (   0.00%)
sequential         0.0000            0.0000            0.0000            0.0000  (   0.00%)
combinational    188.8946          184.1559        2.7231e+04          400.2812  ( 100.00%)
--------------------------------------------------------------------------------------------------
Total            188.8946 uW       184.1559 uW     2.7231e+04 nW       400.2812 uW
1
```

Figure 5: Power disipation of Vedic Multiplier.

in Figure 6. Layout of the Vedic Multiplier without viola-



```
****************************************
Report : area
Design : vedic
Version: O-2018.06-SP3
Date   : Tue Dec  7 21:58:32 2021
****************************************

Library(s) Used:

    NangateOpenCellLibrary (File: /synopsys/Nangate_FreePDK45/Nangate

Number of ports:                           32
Number of nets:                           329
Number of cells:                          304
Number of combinational cells:            301
Number of sequential cells:                 0
Number of macros/black boxes:              0S|
Number of buf/inv:                         42
Number of references:                      15

Combinational area:                375.591997
Buf/Inv area:                       22.344000
Noncombinational area:               0.000000
Macro/Black Box area:                0.000000
Net Interconnect area:      undefined  (Wire load has zero net area)

Total cell area:                   375.591997
```

Figure 6: Area required for the design of Vedic Multiplier.

tions using Cadence Innovous id displayed in Figure 7.

4.1.2 Simulated Results

After we have designed MAC unit with Vedic multiplier we have got the followiung results using ISE Xilinx. Simulation results of Vedic Multiplier is displayed in Figure 8. RTL diagram of Vedic Multiplier is displayed in Figure 9. Simulation results of MAC unit designed with Vedic Multiplier and Half Adder is displayed Figure 10.

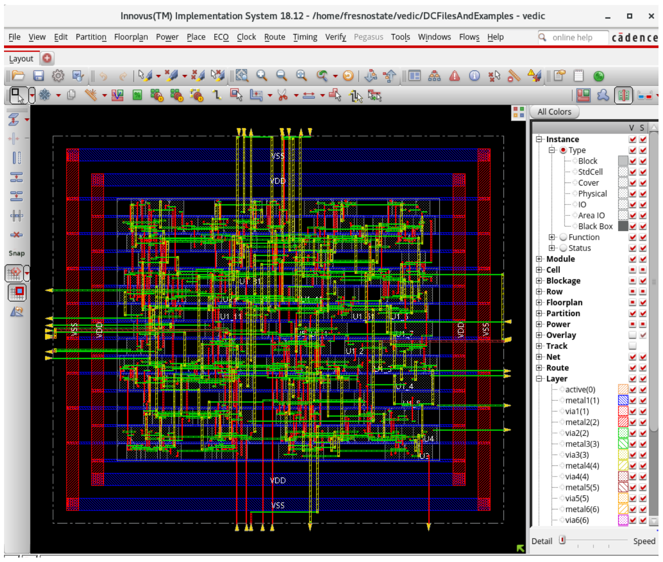
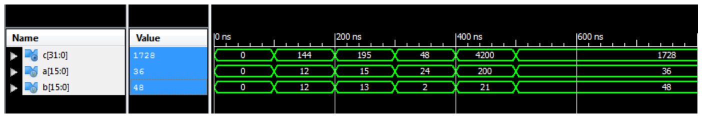


Figure 7: Layout of the design of Vedic Multiplier without violations.



```
Name        Value   |0 ns        |200 ns       |400 ns       |600 ns
c[31:0]     1728     0     144    195    48     4200                 1728
a[15:0]     36       0     12     15     24     200                  36
b[15:0]     48       0     12     13     2      21                   48
```

Figure 8: Simulation results of Vedic Multiplier.

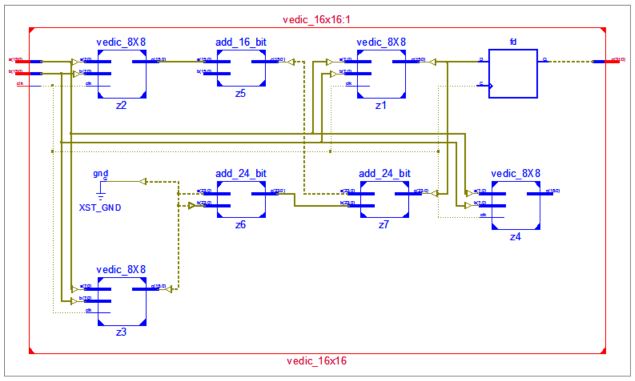


Figure 9: RTL diagram of Vedic Multiplier.

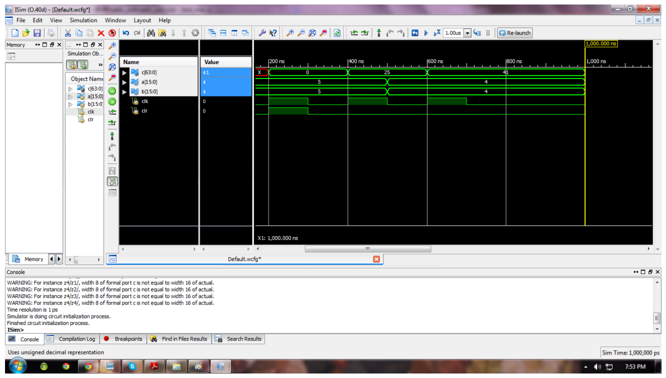
4.2 Analysis

After comparing both the Vedic Multiplier and Radix-2 Booths Multipliers, from the results we have generated is displayed in Figure 11.

5 Conclusion

In digital signal processing, the MAC unit is a crucial component. A new MAC unit was built in this paper using a Vedic multiplier and HCA. It produced more exact and efficient results when compared to the conventional Radix-2 bit booth multiplier with half carry forward adder. As a result, with large inputs, design complexity decreases and flexibility rises, allowing for successful use in a variety of Digital Signal Processing applications.

For, future work Reversible logic gates will be employed to generate multiplier circuits. It consumes less energy than ordinary logic gates.


ISim (O.40d) - [Default.wcfg*]
File Edit View Simulation Window Layout Help
Memory
Name
Value
c[63:0]
a[15:0]
b[15:0]
clk
clr
1,000.000 ns
Default.wcfg*

```
X1: 1,000.000 ns
Console
WARNING: For instance z4/z1/, width 8 of formal port c is not equal to width 16 of actual.
WARNING: For instance z4/z2/, width 8 of formal port c is not equal to width 16 of actual.
WARNING: For instance z4/z3/, width 8 of formal port c is not equal to width 16 of actual.
WARNING: For instance z4/z4/, width 8 of formal port c is not equal to width 16 of actual.
Time resolution is 1 ps
Simulator is doing circuit initialization process.
Finished circuit initialization process.
ISim>
Uses unsigned decimal representation                    Sim Time: 1,000,000 ps
```

Figure 10: Simulation results of MAC unit designed with Vedic Multiplier and Half Adder.

Parameters	Vedic Multiplier	Radix-2 bit Booths Multiplier
LUT	458	764
Slices	308	409
Delay(ns)	8.65	19.18

Figure 11: Comparison of Vedic and Booths Multipliers.

Acknowledgments

The work was performed as a term paper presentation for a course at the California State University, Fresno. As the author, I would like to give a special appreciation to Jin Park for his constant support and guidance during the tenure of this research. He helped me to understand the former research and make this successful.

References

[1] Patil, P. A. & Kulkarni, C. A survey on multiply accumulate unit. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, 1–5 (2018).

[2] Rakesh, S. & Grace, K. S. V. A survey on the design and performance of various mac unit architectures. *2017 IEEE International Conference on Circuits and Systems (ICCS)* 312–315 (2017).

[3] Kaur, A., Kaur, B. & Kaur, R. An efficient multiply accumulate unit design using vedic mathematics algorithm (2020). URL https://issuu.com/irjet/docs/irjet-v7i1171.

[4] Akhter, S. Vhdl implementation of fast nxn multiplier based on vedic mathematic. In *2007 18th European Conference on Circuit Theory and Design*, 472–475 (2007).

[5] Patil, P. A. & Kulkarni, C. Multiply accumulate unit using radix-4 booth encoding. In *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, 1076–1080 (2018).

[6] Kaur, S., Suman & Manna, M. S. Implementation of modified booth algorithm (radix 4) and its comparison with booth algorithm (radix-2) (2013).

[7] S, S., Farooq, F., Krishnan, J., S, R. & Raveendran, A. Implementation of mac by using modified vedic multiplier (2012).

[8] SaiKumar, M. & Samundiswary, P. Design and performance analysis of various adders using verilog (2013). URL https://www.ijcsmc.com/docs/papers/September2013/V2I9201349.pdf.

[9] Gadakh, S. N. & Khade, A. S. Design and optimization of 16×16 bit multiplier using vedic mathematics. *2016 International Conference on Automatic Control and Dynamic Optimization Techniques (ICACDOT)* 460–464 (2016).

[10] Lilly, K., Nagaraj, S., Manvitha, B. & Lekhya, K. Analysis of 32-bit multiply and accumulate unit (mac) using vedic multiplier. In *2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE)*, 1–4 (2020).