VEDIC MULTIPLIER USING CARRY SAVE ADDER

MINI PROJECT REPORT

Submitted to

Visvesvaraya Technological University

BELAGAVI -590 018

by

Prathvi S Kulkarni	**USN: 4SU21EC062**
Praveen T U	**USN: 4SU21EC064**
Rudra Gowda K S	**USN: 4SU21EC071**
Umamaheshwari M Patil	**USN: 4SU21EC095**

Under the guidance of

Mr. Ramachandra

Assistant Professor

in partial fulfilment of the requirements for the award of the degree of

Bachelor of Engineering

Department of Electronics & Communication Engineering

SDM INSTITUTE OF TECHNOLOGY

UJIRE – 574 240

2023-2024

SDM INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

UJIRE -574 240

Department of Electronics and Communication Engineering

CERTIFICATE

Certified that the Mini Project Work titled **'Vedic Multiplier Using Carry Save Adder'** is carried out by **Ms. Prathvi S Kulkarni**, USN: **4SU21EC062**, **Mr. Praveen T U**, USN: **4SU21EC064**, **Mr. Rudra Gowda K S,** USN**: 4SU21EC071**, **Ms. Umamaheshwari M Patil**, USN: **4SU21EC095**, bonafide students of SDM Institute of Technology, Ujire, in partial fulfillment for the award of the degree of **Bachelor of Engineering** in Electronics and Communication Engineering of Visvesvaraya Technological University, Belagavi during the year 2023-2024. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The report has been approved as it satisfies the academic requirements with respect to mini project work prescribed for the said degree.

Mr. Ramachandra
Asst. Professor and Guide

Dr. Madhusudhana K
Head of the Department

ACKNOWLEDGEMENT

We express our deepest gratitude to our guide Mr. Ramachandra, Asst. Professor, Department of Electronics and Communication, for his valuable guidance and encouragement while doing this mini-project work.

We are indebted to Dr. Madhusudhana K, Head of the Department, and Dr. Ashok Kumar T, Principal, for their advice and suggestions at various stages of the work.

We also extend our thanks to the management of SDM Institute of Technology, Ujire, for providing an excellent study environment, reference materials, and laboratory facilities. We remain grateful for the cooperation and help rendered by the teaching and non-teaching staff of the department.

Besides, we sincerely acknowledge the useful comments and assistance given to the mini-project coordinators Dr. Mohan Naik R and Mr. Mahesh D S during the course of this work.

Lastly, we take this opportunity to offer our regards to all of those who have supported us directly or indirectly in completing this mini project work.

Prathvi S Kulkarni
Praveen T U
Rudra Gowda K S
Umamaheshwari M Patil

ABSTRACT

The performance of a DSP processor depends on the efficient multiplier as it is one of the most principal component in various digital circuits. The foresaid technique is for Vedic multiplier design which is based on Urdhva Tiryagbhyam Sutra of Ancient Indian Vedic Mathematics. This project presents modified binary Vedic multiplication using carry save adder. The modified binary Vedic multiplication technique is more efficient in terms of delay. The proposed circuit will be implemented in Verilog HDL. The Xilinx ISE Design Suite is used for circuit synthesis. The simulation will be carried out for 4-bit multiplication operation. The Vedic multiplication method can be extended for a larger bit size. The delay is compared with normal multiplication technique.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

UT	Urdhva Tiryagbhyam
FPGA	Field Programmable Gate Array
ASIC	Application Specific Integrated Circuit
DSP	Digital Signal Processing
CPLD	Complex Programmable Logic Device
CSA	Carry Save Adder
CSA	Carry Select Adder
HDL	Hardware Description Language
ISE	Integrated Software Environment
CLA	Carry Look Ahead Adder
CMOS	Complementary Metal Oxide Semiconductor
HA	Half Adder
FA	Full Adder
KSA	Kogge Stone Adder

CHAPTER 1

INTRODUCTION

1.1 Project Introduction

Multipliers are widely designed for the applications such as Microprocessors, Communication and Digital Signal Processing (DSP). The highly efficient multipliers's demand is increasing with the gradual advancement of microprocessors. For higher order multiplications addition of partial products are extensively used by the adders. Higher order arithmetic operations are crucial to extract the desired output in various real time image and DSP applications. In an efficient a digital circuit, less delay and less power usage are key requirements. Multiplication as an application is the key arithmetic operation and the rapid advancement of fast multiplier circuits has always been a matter of subject for the last couple of decades. Vedic mathematics relied upon DSP operations and also reduced the total processing time as compared with other counterparts. Thus, it is one of the efficient technique for fast multiplication technique.

In ancient India, Vedic mathematics was popularly followed, and it is applied in various mathematical branches. The word Vedic represents 'the storehouse of all knowledge' Vedic mathematics is mainly based on 16 sutras, which were rediscovered in the 20th century. In each sutra, there are certain limitations. Among these 16 Sutras in Vedic mathematics, the Urdhva Tiryagbhyam Sutra is considered a multiplier technique in this paper. Due to the parallel computational approach, the Vedic multiplier approaches the overall speed and power consumption. Unlike traditional multiplication methods taught in modern mathematics, the Vedic multiplier breaks down multiplication into smaller, more manageable steps, making it easier to perform mental arithmetic with large numbers. By utilizing patterns, specific formulae, and digit-wise calculations, the Vedic multiplier enables faster computation and holds relevance in both historical mathematical practices and contemporary applications in computing and digital arithmetic.

'Urdhva Tiryagbhyam Sutra' in Vedic mathematics is generally used for multiplication. The Vedic words Urdhva means vertical, and Tiryagbhyam means crosswise. Therefore, in the vertical and crosswise directions, two binary numbers can be multiplied. Generation of all partial products can be obtained with concurrent addition; by using UT sutra, NxN multiplier architectures can be generated. Fast and accurate operation of digital systems depends on the performance of adders.

Using carry-save addition, the delay can be reduced further. A carry-save adder is a type of digital adder used to efficiently compute the sum of three or more binary numbers. It differs from other digital adders in that it outputs two (or more) numbers, and the answers to the original summation can be achieved by adding these outputs together. Carry Save Adders have several advantages over other types of adders, including reduced propagation delay, lower power consumption, and faster computation. Reconfigurable Field Programmable Gate Arrays (FPGAs) have been gaining in popularity in recent years because it offers improved performance in terms of speed and power over DSP-based and microprocessor-based solutions for many practical designs and a significant reduction in development time and cost over Application Specific Integrated Circuit (ASIC) designs.

Vedic multiplier in real-time applications is highly promising due to its efficiency and reduced computational delay. It can significantly enhance performance in digital signal processing (DSP), image and video processing, cryptography, machine learning, and artificial intelligence by providing faster calculations. Additionally, Vedic multiplier can improve embedded systems' efficiency, extend battery life in devices, and boost communication systems' speed and reliability. In scientific computing, It enables more complex and efficient real-time simulations and computations, making it a valuable asset across various advanced technological fields.

.

CHAPTER 2

LITERATURE REVIEW

2.1 General Introduction

A literature survey is a comprehensive review of existing published research and other relevant sources of information on a particular topic or research question. It involves searching for, collecting, evaluating, and synthesizing relevant literature to identify gaps in the knowledge base and develop a deeper understanding of the topic. It can also help researchers develop research questions, formulate hypotheses, and identify the most appropriate research methods and techniques.

2.2 Literature Survey

1. **Bhavya V, Sunil Kumar H N, Vijaymahantesh, Shreyas R P, Suhas MD, "Modified Binary Vedic Multiplier Using Carry Save Adder" [1].**

   A proposed an efficient novel technique for binary multiplier circuits based on Vedic mathematics. The goal of the project in this paper is to design Implement and analysis of Vedic Multiplier architectures based on Urdhva Tiryagbhyam sutra in Vedic Mathematics. It is proved from the synthesis results that the proposed technique is much efficient in terms of delay. The proposed technique can be extended for a larger bit size. The power analysis has also been done in this work. Reducing the delay is an important requirement for various applications using Carry Save Adder and Vedic Multiplication technique is appropriate for this purpose.

2. **Jatin Yadav, Anupam Kumar, Shaik Shareef, Sandeep Bansal, Navjot Rathour, "Comparative Analysis Of Vedic Multiplier Using Various Adder Architectures" [2].**

   The performance characteristics of Vedic multipliers which are designed using efficient techniques of Vedic mathematics "Urdhva Tiryagbhyam Sutra". This method is based on hierarchical multiplier design which provides an edge in computation compared to other conventional methods. From the comparative analysis, it is clear that Kogge Stone Adder is efficient in terms of power and delay for 45nm and 90nm CMOS technology but seems to be using high power as compared to CSA and CLA. This means with increasing minimum feature size of the transistor, KSA seems to get inefficient in terms power. CSA is more suitable

for applications where area plays an major role. Hence, Vedic Multiplier design provides high efficiency and less complexity in terms of designing the multiplier.

2.3 Summary

In brief, the mentioned research papers focus on reducing the power consumption and delay. It also reviews the performance characteristics of Vedic multipliers which are designed using efficient techniques of Vedic mathematics “Urdhva Tiryagbhyam Sutra”. This method is based on hierarchical multiplier design which provides an edge in computation compared to other conventional methods. In conclusion the Vedic multipliersare much faster than the conventional multipliers. The algorithms of Vedic mathematics aremuch more efficient than of conventional mathematics.

CHAPTER 3

PROBLEM STATEMENT AND OBJECTIVE

3.1 Problem Statement

Conventional multipliers and adders often suffer from large area requirements, time delays, and high-power consumption, making them inefficient for many applications. To overcome these issues, we propose designing a Vedic multiplier using Carry Save Adder (CSA).

3.2 Objective

To reduce the overall time delay in multiplication operations using Vedic multiplier with Carry Save Adder (CSA).

3.3 Summary

The Vedic multiplier simplifies partial product generation, reducing the overall complexity. When combined with the CSA, which efficiently handles carry and sum operations in parallel, this approach significantly decreases computation time and minimizes power consumption. Thus, utilizing a Vedic multiplier with a CSA provides a faster and power saving alternative to traditional methods, making it a superior solution for high performance and power-sensitive applications.

CHAPTER 4

SYSTEM REQUIREMENTS

4.1 Hardware Requirement

4.1.1 FPGA (Spartan-3 XC3S50)

Field-programmable gate arrays (FPGAs) are reprogrammable integrated circuits that contain an array of programmable logic blocks. FPGA chip adoption is driven by their flexibility, hardware-timed speed and reliability, and parallelism.

The Spartan-3 family of Field-Programmable Gate Arrays is specifically designed to meet the needs of high volume, cost-sensitive consumer electronic applications. The Spartan-3 family builds on the success of the earlier Spartan-IIE family by increasing the amount of logic resources, the capacity of internal RAM, the total number of I/O's, and the overall level of performance as well as by improving clock management functions. Because of their exceptionally low cost, Spartan-3 FPGAs are ideally suited to a wide range of consumer electronics applications, including broadband access, home networking, display/projection and digital television equipment. The Spartan-3 family is a superior alternative to mask programmed ASICs.

The XC3S50-PQ208 of Spartan-3 family of Field-Programmable Gate Arrays is specifically designed to meet the needs of high volume, cost-sensitive consumer electronic applications. The eight-member family offers densities ranging from 50,000 to 5,000,000 system gates, as per the Datasheet.

4.2 Software Requirement

4.2.1 Xilinx ISE Design Suite

Xilinx ISE is a discontinued software tool from Xilinx for synthesis and analysis of HDL designs, which primarily targets development of embedded firmware for Xilinx FPGA and CPLD integrated circuit (IC) product families. It was succeeded by Xilinx Vivado. Use of the last released edition from October 2013 continues for in-system programming of legacy hardware designs containing older FPGAs and CPLDs otherwise orphaned by the replacement design tool, Vivado Design Suite. ISE enables the developer to synthesize ("compile") their designs, perform timing analysis, examine RTL diagrams, simulate a design's reaction to different stimuli, and configure the target device with the programmer.

4.2.2 Cadence Genus

The Genus Synthesis Solution is a next-generation RTL synthesis and physical synthesis tool that delivers an up to 10X boost in RTL design productivity with up to 5X faster turn around times. It also delivers tight timing and wirelength correlation to within 5% of place and route. Using the Genus Synthesis Solution, it allows 2X or more reduction in iterations between block-level and unit-level synthesis. In addition, it can achieve an up to 20% reduction in datapath area without any impact on performance. To reduce power consumption, the solution features a comprehensive range of techniques, including timing and physically aware multi-bit flop mapping, hierarchical RTL clock gating, and intelligent one-step use of multiple threshold cell libraries during mapping and optimization.

CHAPTER 5

METHODOLOGY AND IMPLEMENTATION

5.1 Methodology

5.1.1 Urdhva- Triyagbyham Sutra

The main beauty of Vedic multipliers they can perform complex mathematical operations very easily increasing speed. Urdhva Tiryagbhyam is the most generalized sutra for implementation of Vedic Multiplier designs because with increase in number of bits both area and delay increase slowly.

Urdhva Tiryagbhyam Sutra' in Vedic Mathematics is generally used for multiplication. The Vedic words Urdhva means vertical and Tiryagbhyam means crosswise. Therefore, by vertical and crosswise direction two binary numbers can be multiplied. Generation of all partial products can be obtained with the concurrent addition, by using UT sutra NxN multiplier architectures can be generated. Figure 5.1 shows analogy of 2x2 Vedic Multiplication for 2-bit binary numbers and Figure 5.2 shows the analogy of 4x4 Vedic Multiplication for 4- bit binary numbers. The partial product generation is in both vertical and crosswise lines, all the partial products can be generated with concurrent addition.

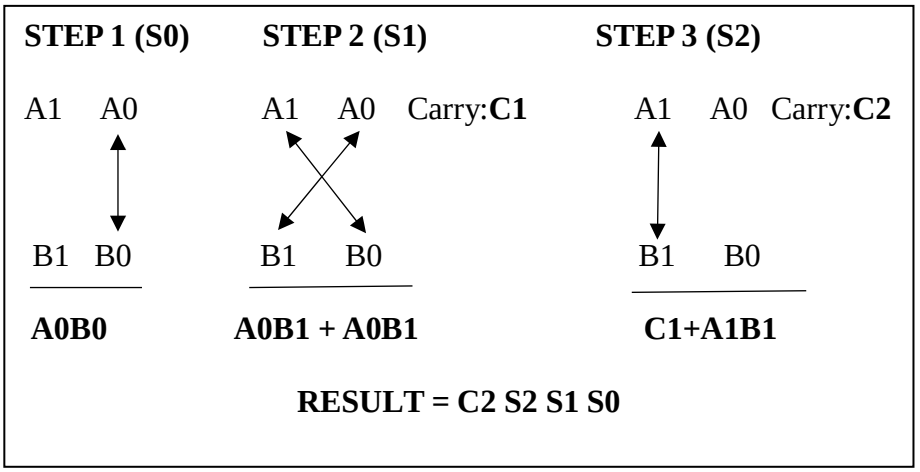

Figure 5.1 2x2 Vedic Multiplication using Urdhva Tiryagbhyam Sutra

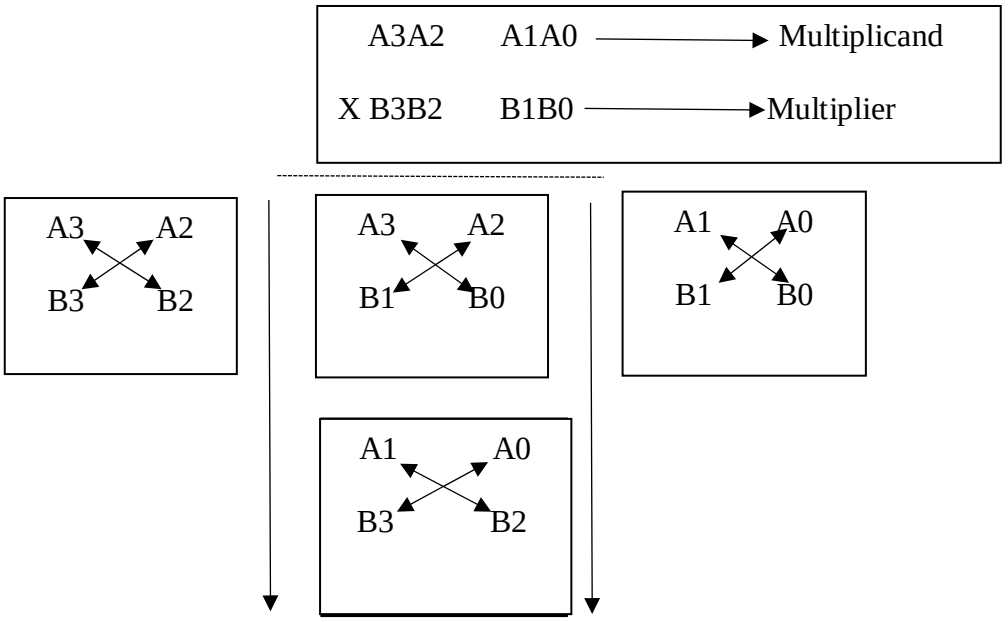

Figure 5.2 4x4 Vedic Multiplication Urdhva Tiryagbhyam Sutra

In order to design 4x4, 8x8 and 16x16 Vedic Multiplier in different modules, 2x2 Vedic Multiplier block is designed, with this block 4x4, 8x8, 16x16 Vedic Multiplier architectures are designed.

5.1.2 VEDIC MULTIPLIER

5.1.2.1 2X2 VEDIC MULTIPLIER

The analogy of 2x2 Vedic Multiplication for 2-bit binary numbers is show in the Figure 5.1, the partial product generation indicated in Figure 5.1 is a vertical and crosswise lines, all the partial products can be generated with concurrent addition.

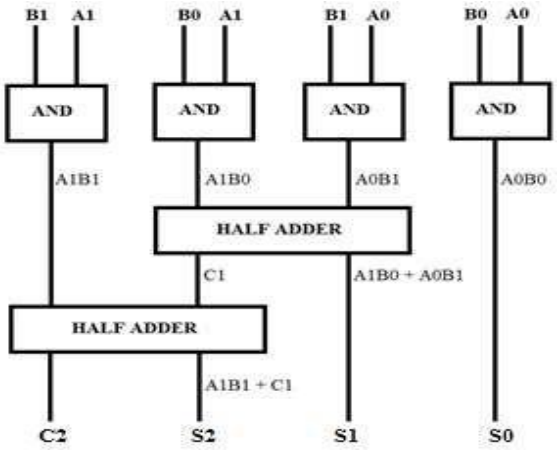

Figure 5.3 Block diagram of 2-bit Vedic multiplier

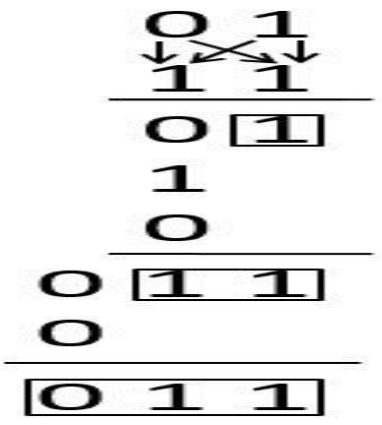

Figure 5.4 2-bit Vedic multiplication

The above Figure 5.3 is the block diagram of 2x2Vedic Multiplier for a 2-Bit binary number containing four AND gate blocks for partial product generation and two half adders for required addition process.

5.1.3 Carry Save Adder

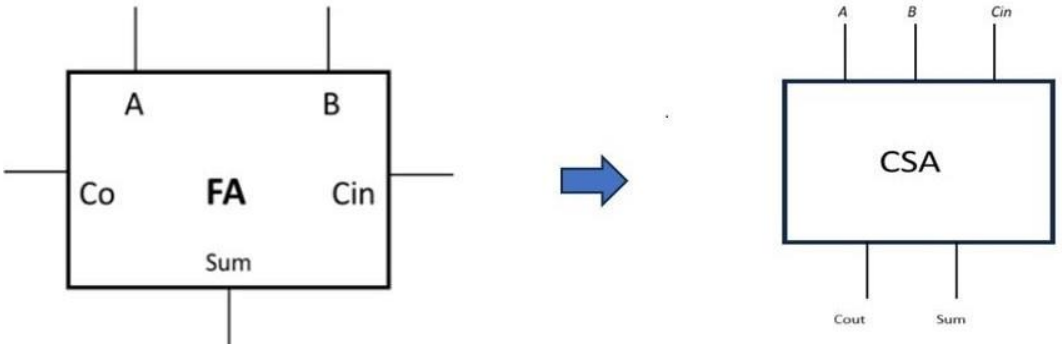

Figure 5.5 Carry Save Adder block same as Full Adder

A carry save adder simply is a full adder with the cin input renamed to z, the z output (the original "answer" output) renamed to s, and the cout output renamed to c. Figure 5.8 showshow n carry save adders are arranged to add three n bit numbers x, y and z into two numbers cand s block diagram for 4-bit carry save adder.

There are many cases where it is desired to add more than two numbers together. The straightforward way of adding together m numbers (all n bits wide) is to add the first two, then add that sum to the next, and so on. This requires a total of m − 1 additions, for a total gate delay of O(m log n). Using carry save addition, the delay can be reduced further still. The idea is to take 3 numbers that to add together, x + y + z, and convert it into 2 numbers c + s such that x + y + z = c + s, and do this in O(1) time. The reason why addition cannot be performed in O(1) time is because the carry information must be propagated. In carry save addition, we refrain from directly passing on the carry information until the very last step. To add three numbers by hand, typically align the three operands, and then proceed column by column in the same fashion that perform addition with two numbers. The three digits ina row are added, and any overflow goes into the next column, when there is some non-zero carry, it is like adding four digits (the digits of x, y and z, plus the carry). Carry Save additionis performed as per the following example.

carry:		1	1	2	1	
x:		1	2	3	4	5
y:		3	8	1	7	2
z:	+	2	0	5	8	7
sum:		7	1	1	0	4

The carry save approach breaks this process down into two steps. The first is to compute the sum ignoring any carries

x:		1	2	3	4	5
y:		3	8	1	7	2
z:	+	2	0	5	8	7
s:		6	0	9	9	4

Each si is equal to the sum of xi + yi + zi modulo 10. Now, separately, compute the carry on a column by column basis

x:		1	2	3	4	5
y:		3	8	1	7	2
z:	+	2	0	5	8	7
c:		1	0	1	1	

After adding the sum and carry values we obtain the final result as computed below.

s:		6	0	9	9	4
c:	+	1	0	1	1	
sum:		7	1	1	0	4

The important point is that c and s can be computed independently, and furthermore, each ci (and si) can be computed independently from all of the other c's (and s's). This achieves original goal of converting three numbers that to add into two numbers that add up to the same sum, and in O(1) time.

x:			1	0	0	1	1
y:			1	1	0	0	1
z:	+		0	1	0	1	1
s:			0	0	0	0	1
c:	+	1	1	0	1	1	
sum:		1	1	0	1	1	1

The same concept can be applied to binary numbers. The above example shows the addition of three 5 bit binary numbers.

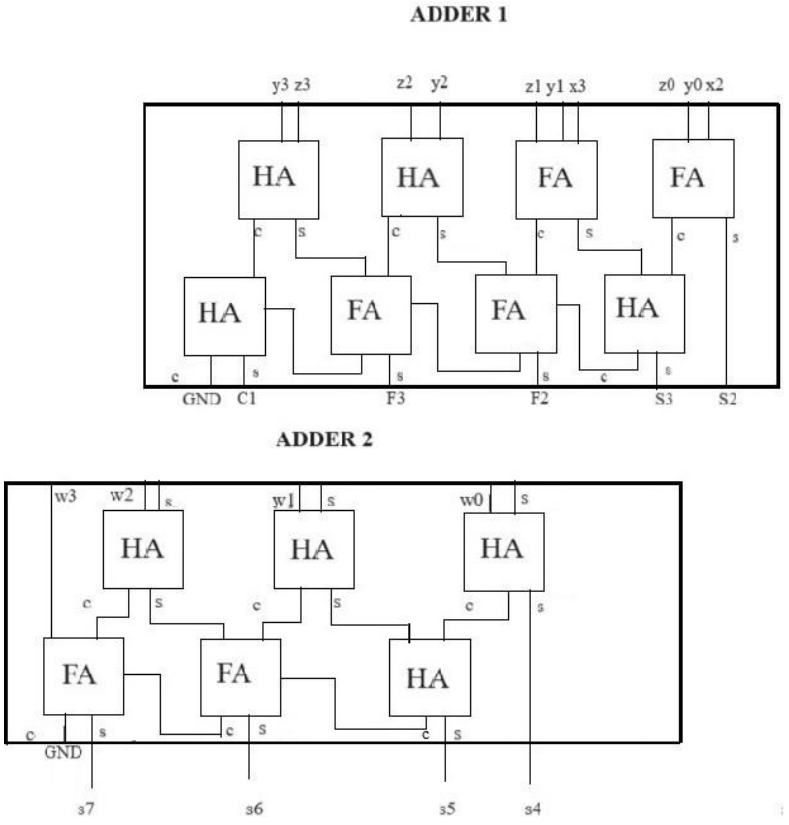

Figure 5.6 Block diagram of 4-bit Carry Save Adder

The diagram in Fig 5.6 represents the modified CSA (Carry Save Adder) block used in a 4x4 Vedic Multiplier. It shows the detailed internal structure of two adders, Adder 1 and Adder 2. Adder 1 Processes initial inputs using HAs and FAs to produce intermediate sums and carries. Adder 2 Takes intermediate outputs from Adder 1 and additional inputs, processes them through HAs and FAs to produce final sum outputs. This structure allows efficient addition operations required in a 4x4 Vedic Multiplier, leveraging this technique to optimize for speed and power consumption.

5.2 Implementation

5.2.1 Design of 4X4 Vedic Multiplier using Carry Save Adder

The Figure 5.7 illustrates a 4x4 Vedic multiplier using four 2x2 Vedic multipliers and two 4-bit carry save adders (CSAs). The 2x2 multipliers compute partial products from the input bits a[3:0] and b[3:0], generating intermediate results. These results are then summed using the CSAs to form the final product. The output bits y[7:0] represent the 8-bit result of the multiplication, with specific bits labelled at each stage of the addition process.

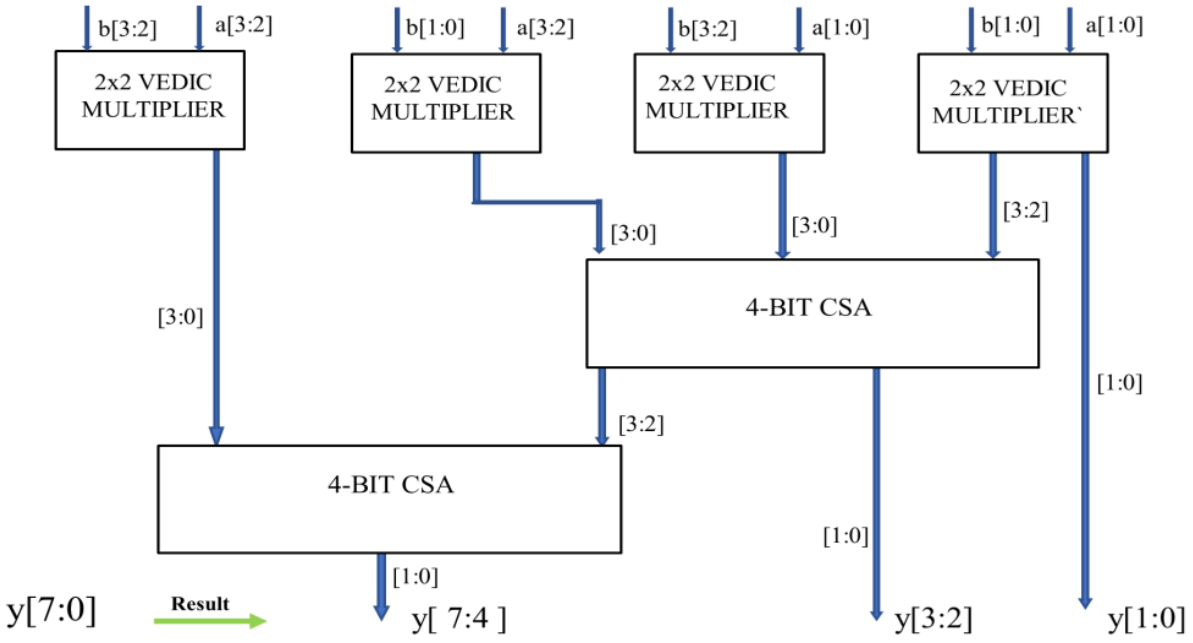

Figure 5.7 Block diagram of CSA based 4x4 Vedic Multiplier

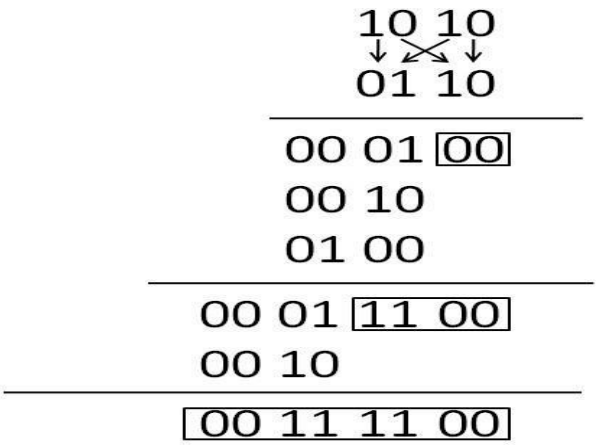

Figure 5.8 Example of 4-bit Vedic Multiplication

CHAPTER 6

RESULTS AND ANALYSIS

6.1 Results

The output of the 4x4 Vedic multiplier executed using the Xilinx ISE tool is shown in Figure 6.1 and Figure 6.2.

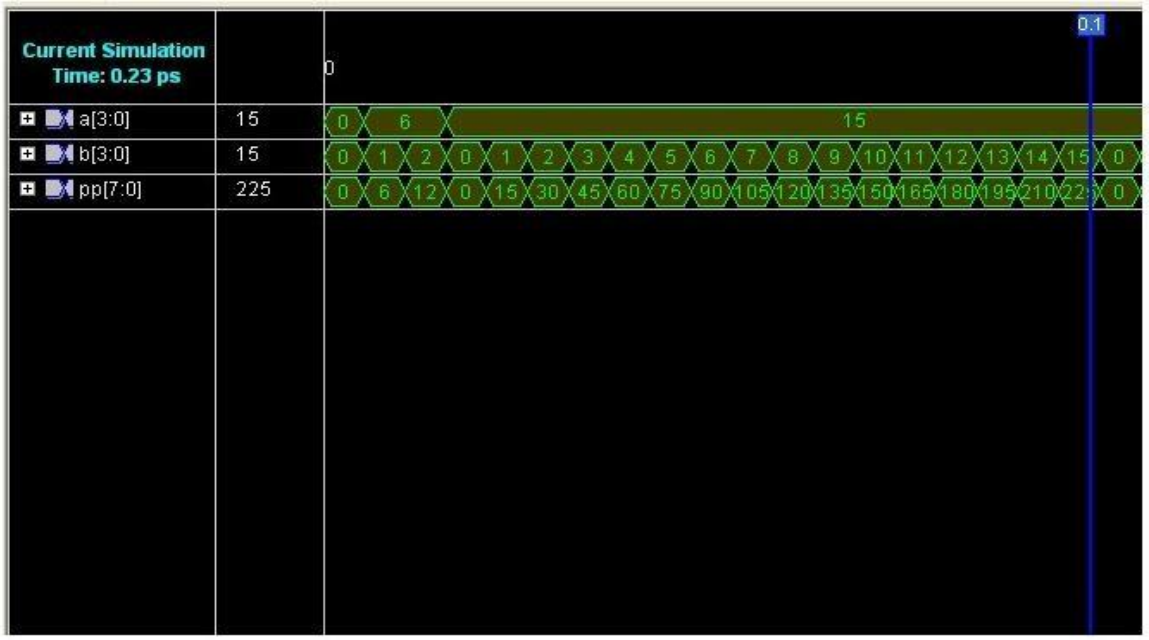

Figure 6.1 Simulation of 4x4 Vedic Multiplier

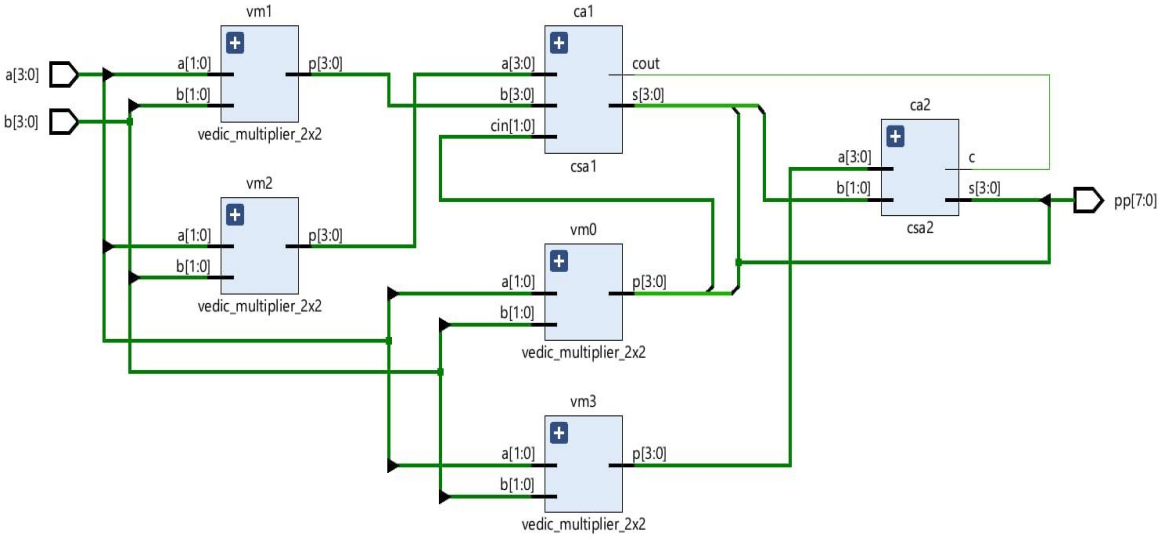

Figure 6.2 Schematic generated by Xilinx

6.2 Analysis

6.2.1 Timing Analysis of 4x4 Vedic Multiplier using CSA

The delay for 4x4 Vedic Multiplier using Carry Save Adder executed using Xilinx ISE is 20.786ns.

```
Timing constraint: Default path analysis
  Total number of paths / destination ports: 808 / 8
--------------------------------------------------------------------------------
Delay:               20.786ns (Levels of Logic = 10)
  Source:            a<0> (PAD)
  Destination:       pp<7> (PAD)

  Data Path: a<0> to pp<7>
                              Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             6   0.821   1.342  a_0_IBUF (a_0_IBUF)
     LUT2:I0->O            2   0.551   1.216  vm1/a0b01 (q<0>)
     LUT4:I0->O            2   0.551   1.216  ca1/fa11/carry1 (ca1/cu1<0>)
     LUT4:I0->O            3   0.551   1.246  ca1/ha21/carry1 (ca1/cu2<0>)
     LUT3:I0->O            3   0.551   1.102  ca1/fa21/carry1 (ca1/cu2<1>)
     LUT4:I1->O            2   0.551   0.877  ca1/Mxor_c_Result1 (c1)
     MUXF5:S->O            2   0.621   0.903  ca2/ha13/Mxor_sum_Result1_f5 (ca2/s1<2>)
     LUT4:I3->O            1   0.551   1.140  ca2/fa21/carry1 (ca2/ca2<1>)
     LUT4:I0->O            1   0.551   0.801  ca2/Mxor_s<3>_xo<1>1 (c3<3>)
     OBUF:I->O                 5.644          pp_7_OBUF (pp<7>)
    ----------------------------------------
    Total                    20.786ns (10.943ns logic, 9.843ns route)
                                      (52.6% logic, 47.4% route)
```

Figure 6.3 Timing Constraint for 4X4 Vedic Multiplier using Carry Save Adder

6.2.2 Timing Analysis of 4x4 Vedic Multiplier using Carry Select Adder

The analysis of the timing delay with Carry Select Adder shows that Vedic Multiplier using Carry Save Adder results in minimum delay.

```
Delay:              21.888ns (Levels of Logic = 13)
  Source:           B<0> (PAD)
  Destination:      product<7> (PAD)

  Data Path: B<0> to product<7>
                              Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             10   0.821   1.329  B_0_IBUF (Mmult_Q0_A<1>_x_B<0>_mand)
     LUT4:I1->O             4   0.551   0.000  Mmult_Q0_Madd_lut<1> (Q0<1>)
     MUXCY:S->O             1   0.500   0.000  Mmult_Q0_Madd_cy<1> (Mmult_Q0_Madd_cy<1>)
     XORCY:CI->O            7   0.904   1.261  Mmult_Q0_Madd_xor<2> (Q0<2>)
     LUT4:I1->O             1   0.551   0.000  Madd_temp2_xor<3>111 (N131)
     MUXF5:I1->O            4   0.360   1.112  Madd_temp2_xor<3>11_f5 (N21)
     LUT2:I1->O             3   0.551   0.933  Madd_temp2_xor<2>11 (temp2<2>)
     LUT4:I3->O             2   0.551   1.216  csa1/rca0/fa1/Mxor_sum_xo<0>31 (csa1/_AUX_2·
     LUT3:I0->O             1   0.551   0.869  csa1/_AUX_2<2>2_SW0 (N61)
     LUT3:I2->O             2   0.551   0.903  csa1/_AUX_2<2>2 (N20)
     LUT4:I3->O             1   0.551   0.827  csa1/_AUX_2<3>21 (N111)
     LUT4:I3->O             1   0.551   0.801  csa1/_AUX_2<3>1 (sum2<3>)
     OBUF:I->O                  5.644          product_7_OBUF (product<7>)
    ----------------------------------------
    Total                     21.888ns (12.637ns logic, 9.251ns route)
                                       (57.7% logic, 42.3% route)
```

Figure 6.4 Timing Constraint for Vedic Multiplier using Carry Select Adder

6.2.3 Comparative Analysis

The comparative analysis of the two multipliers evaluates their performance in terms of time delay for a 4x4 multiplication operation.

Table 6.1 Delay comparison

Sr. No	Design	Time Delay(ns)
1	4x4 Vedic Multiplier using Carry Save Adder	20.786
2	4x4 Vedic Multiplier using Carry Select Adder	21.888

The first design, which we developed, is a 4x4 Vedic Multiplier using a Carry Save Adder and exhibits a time delay of 20.786 nanoseconds. The second design, a 4x4 Vedic Multiplier using Carry Select Adder, is used for comparison and shows a slightly higher time delay of 21.888 nanoseconds. This indicates that our Vedic Multiplier with a Carry Save Adder is faster by approximately 1.102 nanoseconds compared to the Vedic Multiplier using Carry Select Adder. The reduced time delay in our design suggests it is more efficient for applications requiring rapid multiplication operations.

6.2.4 Timing Analysis of Booth Multiplier

The Analysis of the timing delay concludes that Vedic Multiplier has less delay and thus they are treated as high-speed Multipliers as compared to Booth Multiplier.

```
Timing constraint: Default path analysis
  Total number of paths / destination ports: 2397 / 8
--------------------------------------------------------------------------------
Delay:               24.011ns (Levels of Logic = 14)
  Source:            y<1> (PAD)
  Destination:       z<5> (PAD)

  Data Path: y<1> to z<5>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O             24   0.821   2.136  y_1_IBUF (y_1_IBUF)
     LUT4:I0->O             1   0.551   0.869  Mmux_z_mux0008113100_SW0 (N621)
     LUT4:I2->O             1   0.551   1.140  Mmux_z_mux0008113100 (Mmux_z_mux0008113_map
     LUT4:I0->O             2   0.551   0.945  Mmux_z_mux0008113118 (N151)
     LUT4:I2->O             6   0.551   1.342  Mmux_z_mux00081154 (z_mux0009)
     LUT4:I0->O             1   0.551   0.000  Madd__add0005_cy<1>11 (N639)
     MUXF5:I1->O            4   0.360   0.985  Madd__add0005_cy<1>1_f5 (Madd__add0005_cy<1
     LUT4:I2->O             1   0.551   0.000  Mmux_z_mux0008144531 (N643)
     MUXF5:I0->O            1   0.360   0.827  Mmux_z_mux000814453_f5 (Mmux_z_mux0008144_m
     LUT4:I3->O             3   0.551   1.246  Mmux_z_mux000814473 (Mmux_z_mux0008144_map2
     LUT4:I0->O             2   0.551   1.216  Mmux_z_mux000814487 (N221)
     LUT4:I0->O             1   0.551   0.000  Mmux_z_mux0008142721 (N646)
     MUXF5:I1->O            1   0.360   0.801  Mmux_z_mux000814272_f5 (z_5_OBUF)
     OBUF:I->O                  5.644          z_5_OBUF (z<5>)
    ----------------------------------------
    Total                      24.011ns (12.504ns logic, 11.507ns route)
                                        (52.1% logic, 47.9% route)
```

Figure 6.5 Timing Constraint for Booth Multiplier

6.2.5 Comparative Analysis

The comparative analysis of the two multipliers evaluates their performance in terms of time delay for a 4x4 multiplication operation.

Table 6.2 Delay comparison

Sr. No	Design	Time Delay(ns)
1	4x4 Vedic Multiplier using Carry Save Adder	20.786
2	4x4 Booth Multiplier	24.011

The first design, which we developed, is a 4x4 Vedic Multiplier using a Carry Save Adder and exhibits a time delay of 20.786 nanoseconds. The second design, a 4x4 Booth Multiplier, is used for comparison and shows a slightly higher time delay of 24.011 nanoseconds. This indicates that our Vedic Multiplier with a Carry Save Adder is faster by approximately 3.225 nanoseconds compared to the Booth Multiplier. The reduced time delay in our design suggests it is more efficient for applications requiring rapid multiplication operations.

CONCLUSION

The project successfully designed and implemented a Vedic multiplier using a Carry Save Adder (CSA) and simulated it in the Xilinx ISE tool. It was also implemented on an FPGA Spartan kit to validate the design. The performance of this Vedic multiplier was compared with other arithmetic structures like the Carry Select Adder and Booth Multiplier, with a focus on delay metrics.

The simulation and hardware implementation results demonstrate that the Vedic multiplier with CSA achieves significant delay reduction compared to traditional multipliers. The Vedic multiplication algorithm's ability to handle parallel operations, combined with the efficiency of the CSA in minimizing sequential additions, leads to superior speed performance.

When benchmarked against the Carry Select Adder, the Vedic multiplier exhibited a noticeable decrease in delay, affirming its suitability for time-critical applications. Additionally, the comparison with the Booth multiplier highlighted the advantages of the Vedic multiplier's speed and computational efficiency.

The successful implementation of the FPGA Spartan kit further validates the design's practicality and confirms its potential for real-world applications. The Vedic multiplier using CSA proves to be a promising solution for high-performance computing systems where speed, accuracy, and efficient resource utilization are essential.

REFERENCES

[1] Bhavya V, Sunil Kumar H N, Vijaymahantesh, Shreyas R P, Suhas M D, "Modified Binary Vedic Multiplication Using Carry Save Adder", International Journal of Advanced Research in Science, Communication and Technology (IJARSCT) Volume 7, Issue 2, July 2021.

[2] Jatin Yadav, Anupam Kumar, Shaik Shareef, Sandeep Bansal, Navjot Rathour, "Comparative Analysis Of Vedic Multiplier Using Various Adder Architectures", 4th international conference on Intelligent circuits and systems: Journal of physics 2022.

[3] D Khalandar Basha, P Prakash, D M K Chaitanya and K Aruna Manjusha, "RCA - CSA Adder Based Vedic Multiplier", International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 18 (2017) pp. 7603-7613.

[4] CHANDRASHEKARA M N, ROHITH S "Design of 8 Bit Vedic Multiplier Using Urdhva Tiryagbhyam Sutra With Modified Carry Save Adder", 2019 4th International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT-2019), MAY 17th & 18th 2019 978-1-7281-0630-4/19/$31.00©2019 IEEE 116.

APPENDIX A

Verilog Source Code for 4x4 Vedic Multiplier using Carry Save Adder

```
module half_add(a,b,sum,carry);

input a,b;

output sum,carry;

assign sum = a ^ b;

assign carry = a & b;

endmodule

module full_add(a,b,cin,sum,carry);

input a,b,cin;

output sum,carry;

assign sum=a^b^cin;

assign carry =(a&b)|(a&cin)|(b&cin);

endmodule

module csa1(a,b,cin,s,c);

input [3:0] a,b;

input [1:0]cin;

output [3:0] s;

output c;

wire [3:0]s1,cu1;

wire [2:0] cu2;

full_add fa11(a[0],b[0],cin[0],s1[0],cu1[0]);

full_add fa12(a[1],b[1],cin[1],s1[1],cu1[1]);

half_add ha11(a[2],b[2],s1[2],cu1[2]);
```

```
half_add ha12(a[3],b[3],s1[3],cu1[3]);
assign s[0]=s1[0];
half_add ha21(s1[1],cu1[0],s[1],cu2[0]);
full_add fa21(s1[2],cu1[1],cu2[0],s[2],cu2[1]);
full_add fa22(s1[3],cu1[2],cu2[1],s[3],cu2[2]);
assign c = cu1[3] ^ cu2[2];
endmodule
module csa2(a,b,c,s);
input [3:0] a;
input [1:0] b;
input  c;
output [3:0] s;
wire [2:0] s1,ca1;
wire [1:0] ca2;
half_add ha11(a[0],b[0],s1[0],ca1[0]);
half_add ha12(a[1],b[1],s1[1],ca1[1]);
half_add ha13(a[2],c,s1[2],ca1[2]);
assign s[0]=s1[0];
half_add ha21(ca1[0],s1[1],s[1],ca2[0]);
full_add fa21(ca1[1],s1[2],ca2[0],s[2],ca2[1]);
assign s[3]=ca1[2]^a[3]^ca2[1];
endmodule
module vedic_multiplier_2x2(a,b,p);
input [1:0] a, b;
output [3:0] p;
wire a0b0, a0b1, a1b0, a1b1, sum1, carry1;
```

```
assign a0b0 = a[0] & b[0];

assign a0b1 = a[0] & b[1];

assign a1b0 = a[1] & b[0];

assign a1b1 = a[1] & b[1];

half_add ha1 (a0b1, a1b0, sum1, carry1);

assign p[0] = a0b0;

assign p[1] = sum1;

assign p[2] = a1b1 ^ carry1;

assign p[3] = a1b1 & carry1;

endmodule

module vedic_multiplier_4x4(a,b,pp);

input [3:0] a,b;

output [7:0]pp;

wire [3:0]p,q,r,s;

wire [3:0]s1,c3;

vedic_multiplier_2x2 vm0(a[1:0],b[1:0],p);

vedic_multiplier_2x2 vm1(a[1:0],b[3:2],q);

vedic_multiplier_2x2  vm2(a[3:2],b[1:0],r);

vedic_multiplier_2x2  vm3(a[3:2],b[3:2],s);

csa1 ca1(r,q,p[3:2],s1,c1);

csa2 ca2(s,s1[3:2],c1,c3);

assign pp={c3,s1[1:0],p[1:0]};

endmodule
```

Verilog Testbench code for 4x4 Vedic Multiplier

```
module vedic4x4_tb;
reg [3:0]a,b;
 wire [7:0]pp;
vedic_multiplier_4x4 DUT(a,b,pp);
initial
begin
a=4'd0;b=4'd0;
#10 a=4'd6;
#20 a=4'd15;b=4'd15;
#200 $finish;
end
always #10 b=b+4'd1;
endmodule
```

PERSONAL PROFILE

Mr. Ramachandra

Asst. Professor

Project Guide

Mr. Ramachandra received the B.E. degree in Electronics and communication Engineering from Anjuman Engineering College, Bhatkal in the year 2006, and M.Tech in KLE Dr. M S Sheshgiri College of Engineering and Technology, Belagavi in the year 2012.

His subjects of interest include, Microelectronics, MEMS, Embedded system, Analog circuits

Email id: ramachandra@sdmit.in

Contact No: 9742023536

Name: Prathvi S Kulkarni

USN: 4SU21EC062

Address: D/o Shashimouli Kulkarni, Yadwad, Dharwad-581206

Email id: kulkarniprathvi25@gmail.com

Contact No: 9035927825

Name: Praveen T U

USN: 4SU21EC064

Address: S/o Umeshwaraiah, Shivakumar swamy badvane Davanagere-577004

Email id: tupraveen30@gmail.com

Contact No: 6361602165

Name: Rudra Gowda K S

USN: 4SU21EC071

Address: S/o Shanmukhappa, Honnali, Davanagere 577217

Email id: rudragowdaks2003@gmail.com

Contact No: 9110410397

Name: Umamaheshwari M Patil

USN: 4SU21EC095

Address: D/o Mahantesh Patil, Yellapur- 581359, Uttara Kannada

Email id: umamaheshwaripatil.777@gmail.com

Contact No: 9113987855