Design and Implementation Of 32-bit Vedic Multiplier

Impana.C.S, Impana.D, Keerthi Shankar.M.S, Krupa.Y.T, Santhosh Babu.K.C

Department of Electronics and Communication Engineering

P.E.S. College Of Engineering, Mandya, INDIA

keerthishankarms@gmail.com

Abstract

Multipliers form an obligatory component in arithmetic operations. Therefore, in fast-processing systems, the most practical multipliers are required. The essential objective of this work is to make a successful multiplier that uses less on-chip power and delay than conventional multipliers. The Urdhva Tiryakbhyam (Vertically and Crosswise) Sutra, one of the 16 sutras in Vedic mathematics, is the method explained here. This research employs a carry-look-ahead-based carry increment adder to construct a 32x32-bit Vedic multiplier. For the comparison, the 32-bit Array multiplier and 32-bit modified-booth multiplier have been synthesized and implemented. As per the results, it is proven that the parameters considered (delay and power consumption) are reduced in Vedic multiplier than the conventional methods and show that it has better performance. Verilog HDL is utilized to code every one of these parts. After cautiously analyzing the multiplication tasks, Vivado 2023.2 programming is utilized to procure reenactment, combination, and execution results.

Keywords—*Vedic mathematics, Urdhva-Tiryakbhyam Sutra, Carry-increment adder, Verilog HDL, modified-booth multiplier, array multiplier*

I. Introduction

Multiplication is an essential activity in arithmetic operations, which is utilized in Digital Signal Processing (DSP) such as FIR filter, Fast Fourier Transform (FFT), convolution, and ALU (Arithmetic and logical unit) in Microchip. Since Multiplication requests the execution time in most of the DSP applications, there is a prerequisite for a high-speed multiplier. High throughput mathematical operations are fundamental to accomplish in some real-time and picture-handling applications. One of the fundamental operations in such an application is multiplication. The improvement of fast multiplier circuits has been a degree of interest for quite a while. A multiplier considering Vedic Science is one of the fastest and low-power multipliers. Limiting time delay and power usage are especially major essentials for certain applications. Employing this method in the calculation will lessen the intricacy, execution time, power, and many more.

Vedic Mathematics is the oldest known mathematical system. The word ‘Vedic’ is the Sanskrit word ‘Veda’ which implies – the storehouse of all knowledge. This arrangement of math is basic and straightforward. Vedic math was restored from Vedas in 1911 and 1918 by Expert Bharati Krishna Tirthaji Maharaj (1884 - 1960). As shown by his survey, all math depends on sixteen sutras or formulae and thirteen sub-sutras.

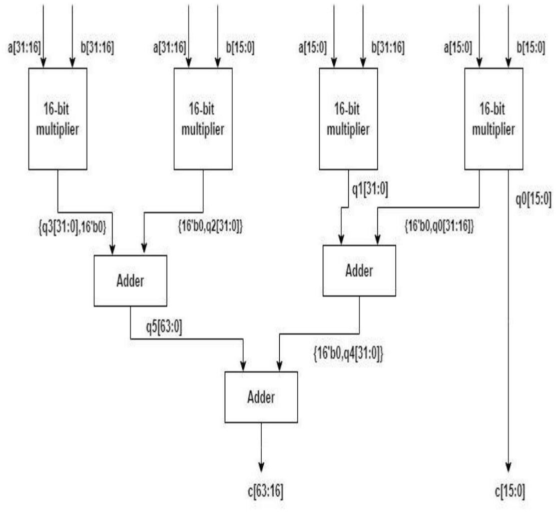
This paper proposes a straightforward 32-bit computerized multiplier utilizing the Urdhva-Tiryakbhyam (Vertically and Crosswise) Sutra of Vedic math. Urdhva-Tiryakbhyam sutra is an overall multiplication formula applicable to all cases of multiplication. Generally, Urdhva-Tiryakbhyam means ‘Vertically and Crosswise’. It relies upon a unique thought through which the generation of all the intermediate results can be done with the simultaneous addition of partial products and their simulation is obtained using the Urdhva-Tiryakbhyam sutra. Two 32-bit binary numbers are multiplied using this sutra. This paper's primary idea is the circuit's power utilization and the proposed engineering's propagation delay.

Array multipliers and Modified-Booth Multipliers are standard methodologies for digital multipliers, suitable for VLSI implementation. This design is executed using Verilog-HDL and simulated with Xilinx Vivado 2023.2. The paper is structured as follows: Section II details the Array and Modified-Booth Multipliers, while Section III discusses the Carry-Increment Adder. Section IV describes the proposed 32-bit multiplier architecture based on Vedic multiplication. The results are discussed in Section V, followed by the conclusion and references in Sections VI and VII, respectively.

II. ARRAY AND MODIFIED BOOTH MULTIPLIER

A. Array Multiplier:

A digital circuit that multiplies two binary values is known as an array multiplier. It chiefly comprises two parts: a network of AND gates to generate partial products and an array of adders to sum up these intermediate products to generate the final product. For all of the multiplicands, the whole multiplier is multiplied using the AND gate to produce the intermediate products. The summation of these fractional items brings about the eventual outcome.



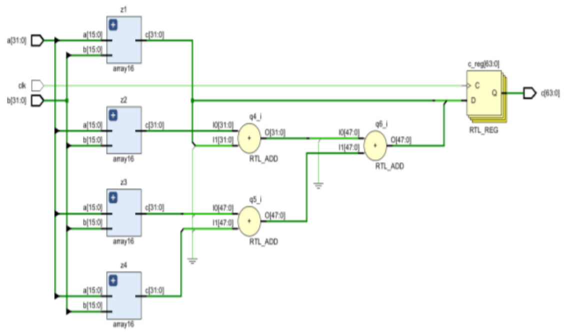


Fig.1 Implementation diagram of Array Multiplier

B. Modified Booth Multiplier:

The Modified-Booth Multiplier is an enhancement of the Booth algorithm used for binary multiplication, focusing on lowering the count of partial-products and shift operations for quicker multiplication. In this algorithm, three bits from the multiplier are analyzed at a time. Depending on the combination of 3-bit binary numbers, explicit operations are carried out on the product register. The operations comprise addition and subtraction of the multiplicand and its shifted versions. After each operation, both the product register and the multiplier are right-shifted by one bit. This method is rehashed until every bit of the multiplier is analyzed, yielding in the eventual outcome.

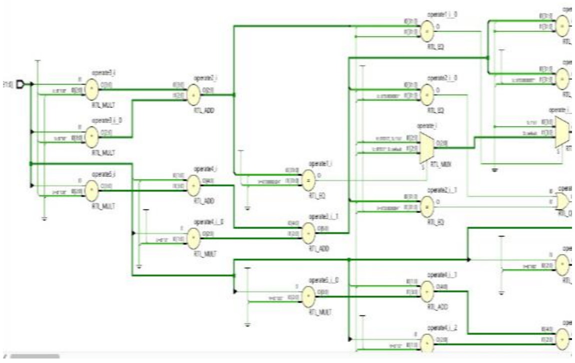


Fig.2 Implementation diagram of Modified-Booth Multiplier

III. CARRY-INCREMENT ADDER

The Carry-Increment Adder (CIA) combines a progressive circuit design with a Carry-Look-Ahead Adder (CLA). The incremental circuitry, constructed using half adders, is consistently connected in a sequential order. The CLA functions by generating two key signals: Propagate and Generate. In the CLA, the Propagate signal represents the sum produced by the Half Adder, while the Generate signal represents the carry output. When the Propagate and Generate signals are determined, the carry is propagated to every bit position. In this design, an 8-bit input can be divided into two 4-bit groups, with the addition performed using a CLA, four Half Adders, and one OR gate. A 16-bit CIA is composed of two 8-bit CLAs and an incremental circuit made up of eight Half Adders followed by an OR gate. The most significant bit (MSB) and the carry output from the second CLA serve as inputs to the OR gate, ultimately producing the final carry output.

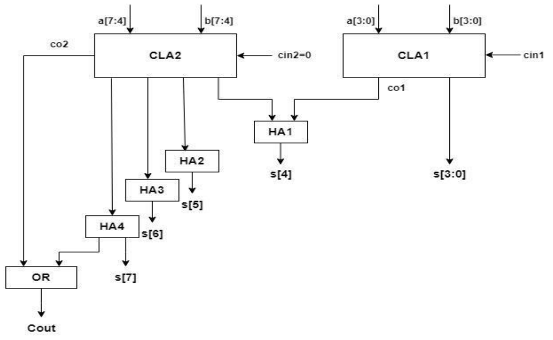


Fig.3 Block diagram of 8-bit CIA using CLA

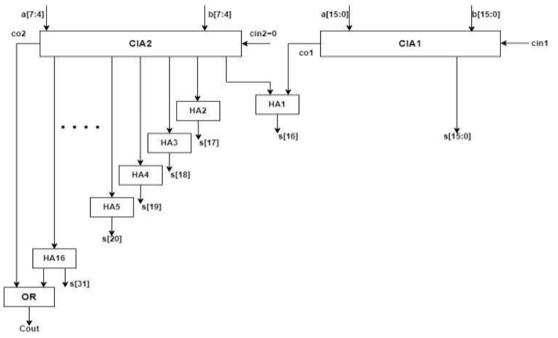


Fig.4 Block Diagram of a 32-bit Carry-Increment Adder (CIA) Utilizing 16-bit CIA Module

PROPOSED METHODOLOGY

The Urdhva Tiryakbhyam Sutra, among the 16 main sutras in Vedic math, has been chosen to be used for the Vedic multiplier because, as we are likely aware, it is appropriate in every scenario of multiplication.

Urdhva-Tiryakbhyam Sutra:

An algorithm called Urdhva-Tiryakbhyam is used to multiply two binary values both horizontally and vertically. Using this procedure, the partial products are counted in simultaneously. This sutra holds true for any number of bits up to N. Compared to other multipliers, this one has the advantage of having latency and an area that rises extremely gradually as the bit count rises. In comparison to array and Modified-Booth multipliers, it not only shortens turnaround times but also speeds up the production process. A 2-bit Vedic multiplier is designed using the Urdhva-Tiryakbhyam Sutra with inputs A[1:0] and B[1:0]. S[3:0] is the result's outcome which is as follows

S[0] = (A[0]*B[0])
S[1] = (A[1]*B[0]) + (A[0]*B[1])
S[2] = (A[1]*B[1])
S[3] = Carry if present

The process of multiplication from right to left is as follows:

1. Start by multiplying the multiplicand and multiplier final bits vertically.
2. After that, add the subsequent bits as indicated and multiply them across.
3. Lastly multiply the first bits vertically to obtain the final product.

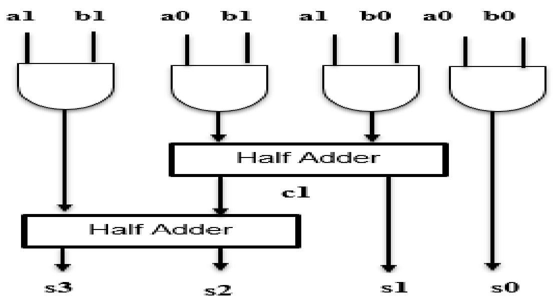


Fig.5 Block diagram of 2-bit Urdhva multiplier

Next, we develop a 4-bit Vedic multiplier using the Urdhva-Tiryakbhyam Sutra with inputs A[3:0] and B[3:0]. S[7:0] is the result's output line. We'll foster the accompanying conditions for the results in light of the thinking behind a 2-bit Vedic multiplier:

S[0] = (A[0]*B[0])

S[1] = (A[0]*B[1]) + (A[1]*B[0])

S[2] = (A[2]*B[0]) + (A[0]*B[2]) + (A[1]*B[1])

S[3] = (A[3]*B[0]) + (A[0]*B[3]) + (A[2]*B[1]) + (A[1]*B[2])

S[4] = (A[1]*B[3]) + (A[3]*B[1]) + (A[2]*B[2])

S[5] = (A[3]*B[2]) + (A[2]*B[3])

S[6] = (A[3]*B[3])

S[7] = Carry if present

Utilizing the information sources A[15:0] and B[15:0], an 8-bit Vedic multiplier is made with the rationale of the Urdhva-Tiryakbhyam Sutra. S[31:0] is the outcome's result line. We create conditions for each total piece, understand the rationale of a 4-bit Vedic multiplier, and utilize extraordinary adders (four input adders) and standard adders, much like in a 4-digit multiplier.

By grouping the 16-bit multiplicands into two 8-bit sections each, A[7:0], B[7:0], and A[15:8], B[15:8], we may expand this to a **16-bit multiplier** with the inputs A[15:0] and B[15:0]. Now that we have these, we can use the 2-bit Urdhva Tiryakbhyam sutra logic to obtain partial products. CIA (base CLA) is used to add these partial products, yielding the final (S[31:0]) 32-bit product.

As a result, two 16-bit Vedic multipliers are used to construct a 32-bit Vedic multiplier, with each 32-bit multiplicand divided into two 16-bit segments. A[15:0], B[15:0], and A[31:16], B[31:16]. Utilizing this, we presently apply the 2-bit Urdhva-Tiryakbhyam sutra rationale, and the outcome is the fractional items shown in Fig.6 beneath. CIA(base CLA) is utilized to join these incomplete items to make the last (S[63:0]) 64-bit item.

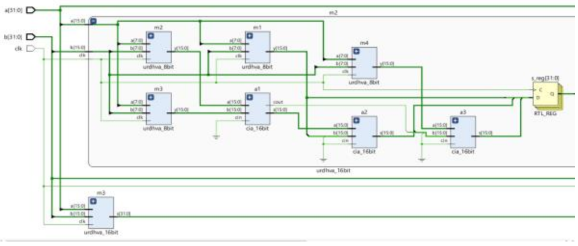


Fig.6 Implementation diagram of 16-bit Urdhva multiplier

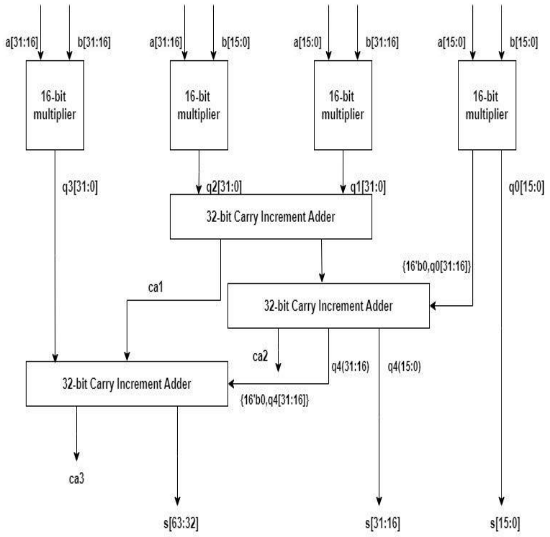


Fig.7 Block diagram of 32-bit Urdhva Tiryakbhyam Sutra

IV. RESULTS

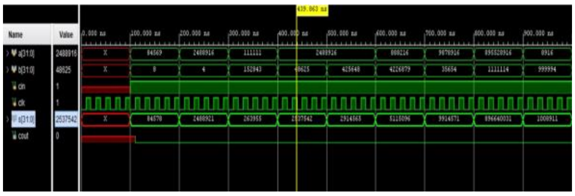


Fig.8 Output waveform of 32-bit CLA

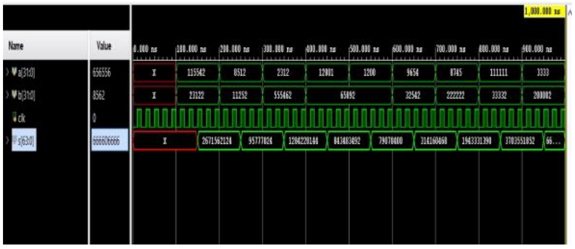
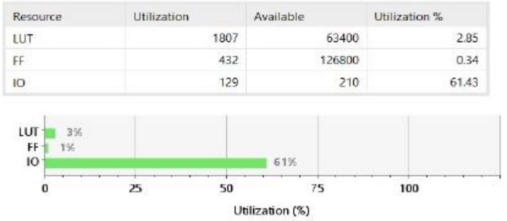
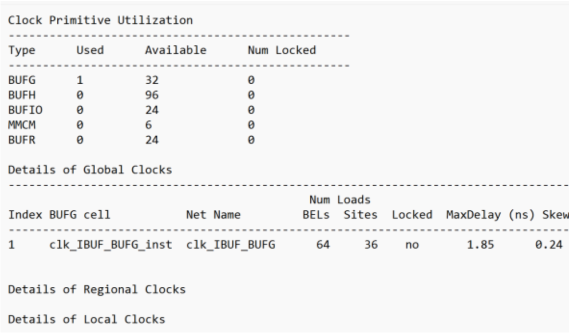


Fig.9 Output waveform of 32-bit Urdhva multiplier



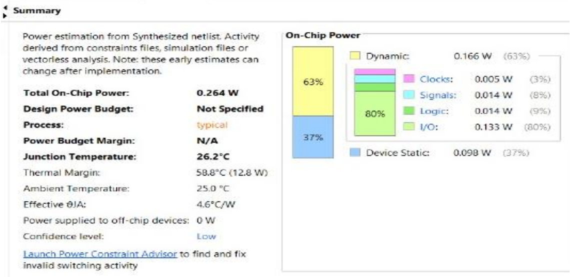
Resource	Utilization	Available	Utilization %
LUT	1807	63400	2.85
FF	432	126800	0.34
IO	129	210	61.43

Fig.10 LUTs count of 32-bit Urdhva multiplier



```
Clock Primitive Utilization
----------------------------------------------------
Type      Used      Available      Num Locked
----------------------------------------------------
BUFG      1         32             0
BUFH      0         96             0
BUFIO     0         24             0
MMCM      0         6              0
BUFR      0         24             0

Details of Global Clocks
------------------------------------------------------------------------------
                                            Num Loads
Index BUFG cell           Net Name        BELs  Sites  Locked  MaxDelay (ns) Skew
------------------------------------------------------------------------------
1     clk_IBUF_BUFG_inst  clk_IBUF_BUFG     64    36     no        1.85       0.24


Details of Regional Clocks

Details of Local Clocks
```

Fig.11 Delay of 32-bit Urdhva multiplier



Summary

Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

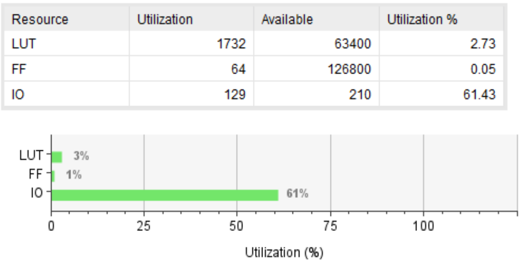
Total On-Chip Power:	**0.264 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**26.2°C**
Thermal Margin:	58.8°C (12.8 W)
Ambient Temperature:	25.0 °C
Effective θJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

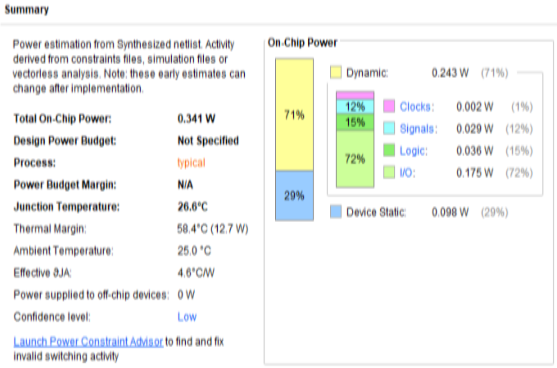
Dynamic:	0.166 W	(63%)
Clocks:	0.005 W	(3%)
Signals:	0.014 W	(8%)
Logic:	0.014 W	(9%)
I/O:	0.133 W	(80%)
Device Static:	0.098 W	(37%)

Fig.12 Total on-chip power of 32-bit Urdhva multiplier



Resource	Utilization	Available	Utilization %
LUT	1732	63400	2.73
FF	64	126800	0.05
IO	129	210	61.43

Fig.13 LUTs count of 32-bit Array multiplier



Summary

Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

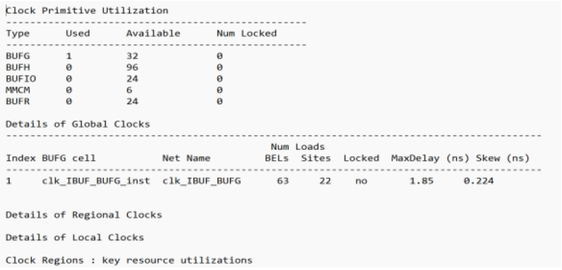
Total On-Chip Power:	**0.341 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**26.6°C**
Thermal Margin:	58.4°C (12.7 W)
Ambient Temperature:	25.0 °C
Effective ϑJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

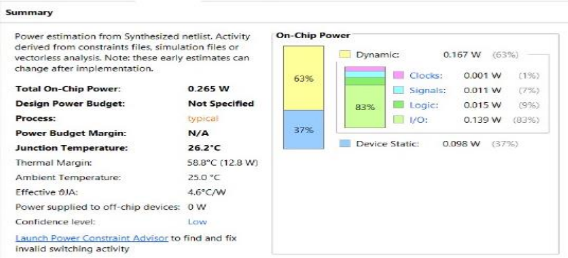
Dynamic:	0.243 W	(71%)
Clocks:	0.002 W	(1%)
Signals:	0.029 W	(12%)
Logic:	0.036 W	(15%)
I/O:	0.175 W	(72%)
Device Static:	0.098 W	(29%)

Fig.14 Total on-chip power of 32-bit Array multiplier



```
Clock Primitive Utilization
----------------------------------------------------
Type      Used      Available      Num Locked
----------------------------------------------------
BUFG      1         32             0
BUFH      0         96             0
BUFIO     0         24             0
MMCM      0         6              0
BUFR      0         24             0

Details of Global Clocks
-------------------------------------------------------------------------------------
                                            Num Loads
Index BUFG cell           Net Name        BELs  Sites  Locked  MaxDelay (ns) Skew (ns)
-------------------------------------------------------------------------------------
1     clk_IBUF_BUFG_inst  clk_IBUF_BUFG     63    22     no        1.85       0.224


Details of Regional Clocks

Details of Local Clocks

Clock Regions : key resource utilizations
```

Fig.15 Delay of 32-bit Array multiplier



Summary

Power estimation from Synthesized netlist. Activity derived from constraints files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

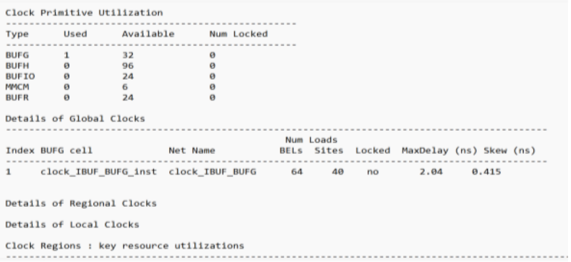
Total On-Chip Power:	**0.265 W**
Design Power Budget:	**Not Specified**
Process:	typical
Power Budget Margin:	**N/A**
Junction Temperature:	**26.2°C**
Thermal Margin:	58.8°C (12.8 W)
Ambient Temperature:	25.0 °C
Effective θJA:	4.6°C/W
Power supplied to off-chip devices:	0 W
Confidence level:	Low

Launch Power Constraint Advisor to find and fix invalid switching activity

On-Chip Power

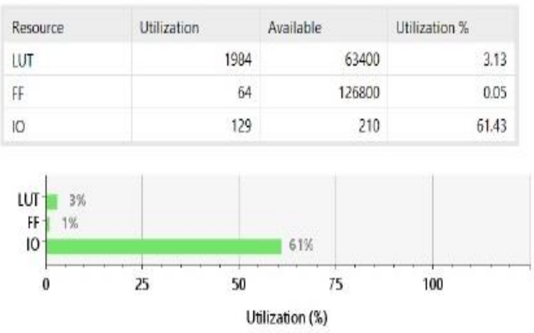
Dynamic:	0.167 W	(63%)
Clocks:	0.001 W	(1%)
Signals:	0.011 W	(7%)
Logic:	0.015 W	(9%)
I/O:	0.139 W	(83%)
Device Static:	0.098 W	(37%)

Fig.16 Total on-chip power of 32-bit Modified Booth



```
Clock Primitive Utilization
----------------------------------------------------
Type      Used      Available      Num Locked
----------------------------------------------------
BUFG      1         32             0
BUFH      0         96             0
BUFIO     0         24             0
MMCM      0         6              0
BUFR      0         24             0

Details of Global Clocks
---------------------------------------------------------------------------------------
                                                Num Loads
Index BUFG cell             Net Name          BELs  Sites  Locked  MaxDelay (ns) Skew (ns)
---------------------------------------------------------------------------------------
1     clock_IBUF_BUFG_inst  clock_IBUF_BUFG     64    40     no        2.04       0.415


Details of Regional Clocks

Details of Local Clocks

Clock Regions : key resource utilizations
-------------------------------------------------------------------------------------------
```

Fig.17 Delay of 32-bit Modified Booth multiplier



Resource	Utilization	Available	Utilization %
LUT	1984	63400	3.13
FF	64	126800	0.05
IO	129	210	61.43

Fig.18 LUTs count of 32-bit Modified Booth multiplier

Table.1 Comparison table

	Urdhva Tiryakbhyam multiplier	Modified Booth Multiplier	Array Multiplier
Power(W)	0.264	0.265	0.341
Delay(ns)	1.85	2.04	1.85
LUTs	1807	1984	1732

V. CONCLUSION

From the above results and its comparison we can say that the Urdhva multiplier is superior. However delay and LUTs count in Array multipliers are less, and power consumption comparatively is high.

In the modified-booth multiplier, power consumption is almost the same as the Urdhva multiplier but delay and LUTs are pretty high. Since our main concentration is on delay and power consumption, on comparing these two parameters we can easily conclude that the Urdhva Tiryakbhyam Sutra gives better performance. Also, the fact that Vedic multipliers are implemented using CIA rather than CLA or RCA, improves its efficiency much more.

REFERENCES

[1]Akshay Savji, Shruti Oza, "Design and Implementation of Vedic Multiplier", International Journal of Recent Technology and Engineering (IJRTE), ISSN: 2277-3878 (Online), Volume-8 Issue-6, March 2020.

[2]Krishnaveni D, Umarani TG, "VLSI Implementation of Vedic multiplier with reduced delay", International Journal of Advanced Technology & Engineering Research (IJATER) National Conference on Emerging Trends in Technology (NCET-Tech), Volume 2, 4 July 2012.

[3]Siba Kumar Panda, Ritisnigdha Das, S K Saifur Raheman, Tapas Ranjan Sahoo, "VLSI Implementation of Vedic Multiplier Using Urdhva–Tiryakbhyam Sutra in VHDL Environment", IOSR Journal of VLSI and Signal Processing (IOSR-JVSP), Volume 5, Issue 1, Ver. III (Jan - Feb. 2015), PP 17-24, e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197.

[4]Amrita Nanda, Shreetam Behera, "Design and Implementation of Urdhva Tiryakbhyam Based Fast 8×8 Vedic Binary Multiplier", International Journal of Engineering Research & Technology (IJERT) IJERT ISSN: 2278-0181, Vol. 3 Issue 3, March - 2014.

[5]Garima Rawat, Khyati Rathore, Siddharth Goyal, Shefali Kala, Poornima Mittal, "Design and Analysis of ALU: Vedic Mathematics Approach", IEEE, 15-16 May 2015.

[6]Prabir Saha, Arindam Banerjee, Partha Bhattacharyya, Anup Dandapat, "High-speed ASIC design of complex multiplier using Vedic Mathematics", IEEE, 14-16 January 2011, lIT Kharagpur.

[7]J M Rudagi, Vishwanath Ambli, Vishwanath Munavalli, Ravindra Patil, Vinaykumar Sajjan, "Design and implementation of efficient multiplier using Vedic Mathematics", 3rd International Conference on Advances in Recent Technologies in Communication and Computing (ARTCom 2011), 14-15 November 2011.

[8] R Uma, Vidya Vijayan, M Mohanapriya, Sharon Paul, "Area, Delay and Power Comparison of Adder Topologies", International Journal of VLSI design & Communication Systems (VLSICS) Vol.3, No.1, February 2012.

[9] B.Ratna Raju, D.V.Satish, "A High Speed 16*16 Multiplier Based On Urdhva Tiryakbhyam Sutra", International Journal of Science Engineering and Advanced Technology, IJSEAT, Vol 1, Issue 5, October - 2013, ISSN 2321-6905.

[10] Neha Varshney, Greeshma Arya, "Design and Execution of Enhanced Carry Increment Adder using Han-Carlson and KoggeStone adder Technique", Proceedings of the Third International Conference on Electronics Communication and Aerospace Technology [ICECA 2019] IEEE Conference Record # 45616; IEEE Xplore ISBN: 978-1-7281-0167-5

Biographies and Photographs

Impana C S, Final year Student at PES College of Engineering, Mandya. She is pursuing a Bachelor of Engineering (BE) degree specializing in Electronics and Communication Engineering. Her areas of interest include VLSI design, Embedded Systems, Digital electronics, and Vedic math.

Impana D, Final year Student at PES College of Engineering, Mandya. She is pursuing a Bachelor of Engineering (BE) degree specializing in Electronics and Communication Engineering. Her areas of interest include Verilog HDL, Embedded Systems, Digital electronics, and Vedic math.

Keerthi Shankar M S, Final year Student at PES College of Engineering, Mandya. She is pursuing a Bachelor of Engineering (BE) degree specializing in Electronics and Communication Engineering. Her areas of interest include Digital electronics, Embedded Systems, Data Science, and Vedic math.

Krupa Y T, Final year Student at PES College of Engineering, Mandya. She is pursuing a Bachelor of Engineering (BE) degree specializing in Electronics and Communication Engineering. Her areas of interest include Digital electronics, Embedded systems, and Vedic math.

Santhosh Babu K C, received a Bachelor of Engineering (B.E) degree in Electronics and Communication Engineering from PES College Of Engineering, Mandya, and a Master of Technology (M.Tech) from SJBIT, Bengaluru. With experience of 13+ years, he is also pursuing his Ph.D in the field of FINFET Technology. His research interests include VLSI design, FINFET, and Vedic math. He is currently working as an Assistant Professor at PES College of Engineering, Mandya.