IJRAR.ORG E-ISSN: 2348-1269, P-ISSN: 2349-5138

INTERNATIONAL JOURNAL OF RESEARCH AND ANALYTICAL REVIEWS (IJRAR) | IJRAR.ORG

An International Open Access, Peer-reviewed, Refereed Journal

VLSI Implementation of 32 bit Vedic Multiplier usingUrdhva Tiryakbhyam Sutra in Cadence Software.

[1]M.Sowjanya, [2] Dr.P.Vijayi Bhaskar

[1]Associate Professor, [2]Professor

[1]Department of Electronics and Communication Engineering,

[1]Geethanjali College of Engineering and Technology, Hyderabad, India

Abstract—**Utilising vedic sutras from classical vedic mathematics, the proposed project effort specifies a particular binary vedic multiplier. Performance has a crucial role for operational speed. "Multiplier" components are necessary for any processor or other computing device. We can assess its performance by doing a few multiplications within a specific time frame. The device's efficient construction, quick processing, efficient use of power and multiplier computation technique thus determine how effective it is. The recommended vedic multiplier is designed and implemented using Verilog HDL. The software Cadence is used for HDL simulation and circuit synthesis. Four bit, eight bit, sixteen bit, and thirty-two bit multiplication operations have all been simulated. The simulation's performance data is only shown for the 32-bit binary vedic multiplier technique.**

***Key Terms*— Verilog HDL, the Vedic multiplier, the Urdhva Tiryakbhyam sutra, Cadence, Simulation, Synthesis, and Layout**

1. INTRODUCTION

The two main performance criteria for gadgets are power consumption and operational speed. The effectiveness of the internal CPU determines how well these devices perform. Multipliers are one of the most crucial parts of any processor, such as microprocessors, digital signal processors, microcontrollers, and other pieces of computer hardware. Multiplication is a fundamental task in arithmetic operations due to the computationally intensive nature of multiply and accumulate (MAC) and inner product, which are frequently used in many digital signal processing (DSP) applications like convolution, fast Fourier transform (FFT), filtering, and the arithmetic and logic unit (ALU) of microprocessors

The power, chip area utilization, and delay ofa multiplier are all improved by using the Urdhva Tiryakbhyam sutra from the Vedas. Vedic multiplier has a reliable structure and is a practical method for better performance of multipliers.

SOFTWARE REQUIREMENTS

- Cadence

2. LITERATURE SURVEY

R. Anitha used a Vedic multiplier in cadence (45nm technology) to construct discrete linear convolution architecture in November 2016. The design required 52% less area and 71.234% a smaller amount of power as compared to the conventional approach

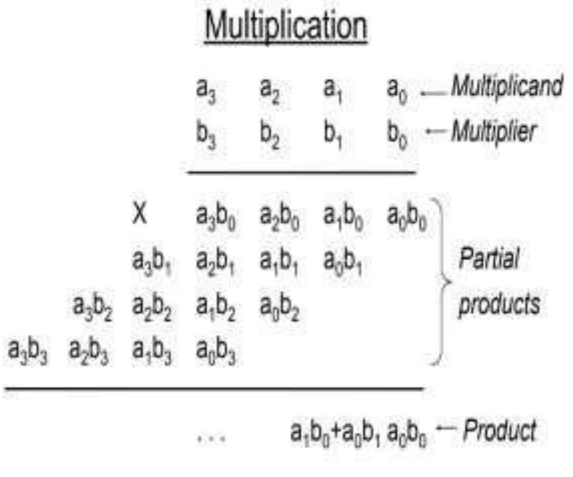
In May 2016, Nitesh Kumar developed an approximate multiplier using the Urdhva tiryakbhyam sutra of Vedic mathematics. The design was finished using Xilinx 14.1. A divide and conquer tactic was suggested to minimize hardware and time complexity by 20 to 30% when compared to the prior approach

The G. Challa Ram Vedic multiplier's performance was evaluated in 2016 by comparing it to the array multiplier architecture in terms of delay, memory utilization, and power consumption. It was discovered that utilizing a Vedic multiplier rather than an array multiplier can reduce the time of multiplication as the number of bits increases.

In November 2015, B. Keerthi Priya and colleagues created a 4-bit multiplier in cadence virtuoso (45nm technology) utilizing GDI as well as tweaked GDI techniques. revised on February 6, 2017 GDI Power consumption and delay were decreased by 80% and 37.6%, respectively, with the addition of the carry saving adder and the improved GDI. They were also decreased by 53.9% and 75.0%, respectively

3. CONVENTIONAL MULTIPLIERS

A multiplier's primary job is to multiply two binary values with different bit sizes in binary. In the first stage of the binary multiplier process, the bit-wise multiplication, which is comparable to ANDing two binary values, is utilized to create the partial product terms. The result of the binary product is obtained in the following step by adding each column's partial product terms. The multiplier's logic circuit design changes depending on the bit size, and its complexity rises as bit size does. When the twobits that need to be multiplied are provided as inputs with different bit sizes, the AND gate functions control it. The multiplier's logic circuit design changes depending on the bit size, and its complexity rises as bit size does. When the two bits that need to be multiplied are provided as inputs with different bit sizes, the AND gate functions control it. If the matching multiplier bit is set to 1, partial products are copies of the multiplicand; otherwise, they are all zeros. The leftmost bit of each subsequent partial product is shifted one bit position.



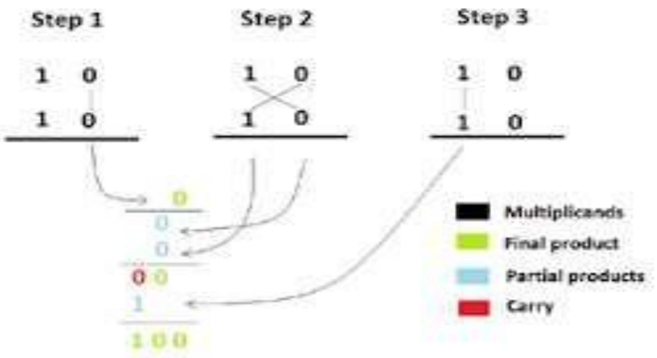
4. VEDIC MULTIPLIER

Vedic algorithms have the unique ability to reduce complex problems to simple calculations, leading to quicker solutions. These benefits of Vedic mathematics are proven to considerably help all areas of mathematics, including arithmetic, calculus, geometry, and computing. As a result, creating a multiplier using Vedic mathematics yields incredible results. It presents a Vedic multiplier that is faster than conventional multipliers and shows how straightforward it is to implement Vedic algorithms on hardware through its FPGA implementation.

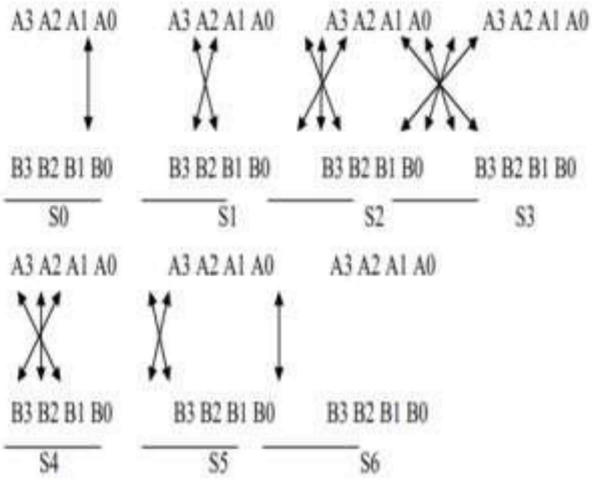
URDHVA TIRYAKBHYAM SUTRA

The "Urdhva Tiryakbhyam" sutra, an algorithm, serves as the foundation for the Vedic multiplier. To multiply two given integers in a decimal number system, this well-known sutra hastraditionally been used. The same idea is applied in this work and tailored to the binary number system to make the suggested algorithm compatible with the digital hardware. It is an all-purpose multiplication formula that works with any multiplication. “It basically means "in a vertical and crosswise direction ". It is a revolutionary concept that enables the creation of all partial products at once and their addition. The technique works when multiplying any two numbers with bit lengths equal to n. Because the partial products and their sums are calculated simultaneously, the multiplier is independent of the processor's clock frequency. The advantage of the multiplier based on this sutra over other conventional multipliers is that gate latency and area rise very slow as the number of bits increases.

ALGORITHM FOR 2x2 VEDIC MULTIPLICATION



ALGORITHM FOR 4x4 VEDIC MULTIPLICATION



Step1 : S0 = A0*B0

Step2: S1 = A1*B0+A0*B1

Step3: S2 = A2*B0+A0*B2+A1*B1

Step4: S3 = A3*B0+A0*B3+A2*B1+A1*B2

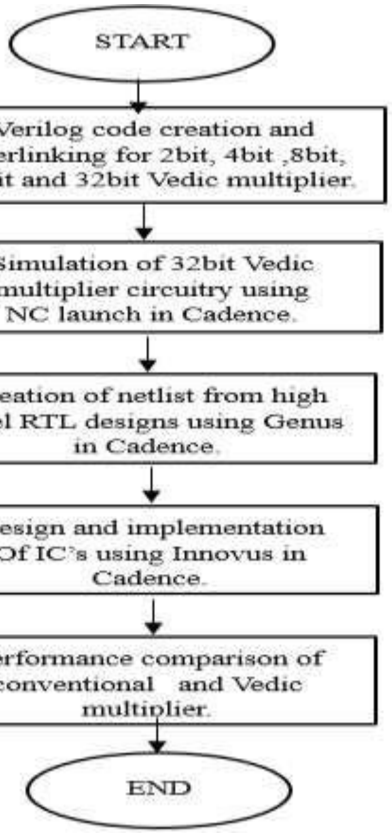
Step5: S4 = A3*B1+A1*B3+A2*B2

Step6: S5 = A3*B2+A2*B3

Step7: S6 = A3*B3

5. METHODOLOGY

5.1 FLOWCHART


START
Verilog code creation and interlinking for 2bit, 4bit ,8bit, 16bit and 32bit Vedic multiplier.
Simulation of 32bit Vedic multiplier circuitry using NC launch in Cadence.
Creation of netlist from high level RTL designs using Genus in Cadence.
Design and implementation Of IC's using Innovus in Cadence.
Performance comparison of conventional and Vedic multiplier.
END

Step 1:- Create a Verilog code for 32 bit Vedic Multiplier

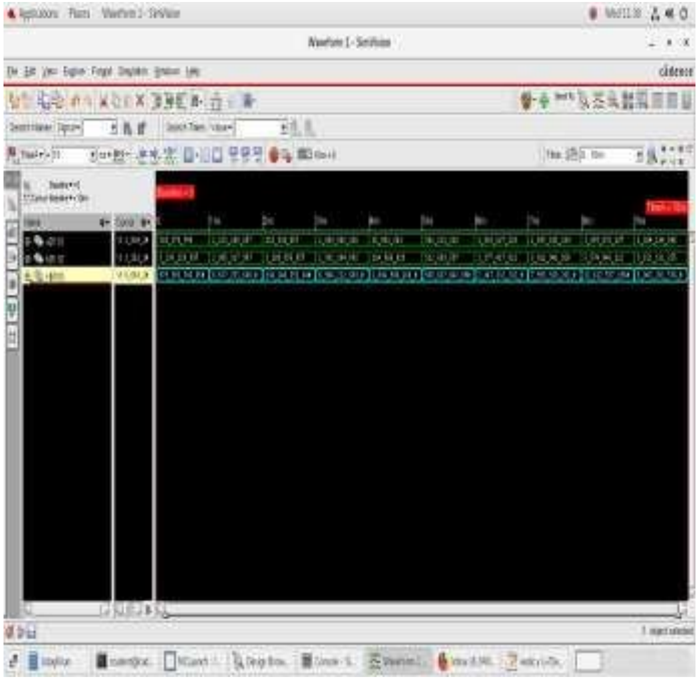
Step 2:- NC launch is a powerful and versatile tool developed by Cadence that is used for launching and managing simulations of electronic circuits. This tool is used extensively in the electronics industry for its ability to manage and control the simulation of complex electronic designs.

Step 3:- Genus is a high-level RTL synthesis tool developed by Cadence, which is used to automatically generate gate-level net lists from Register-Transfer Level (RTL) designs. The tool's great performance, precision, and sophisticated optimization capabilities are key factors in its widespread use in the electronics sector.

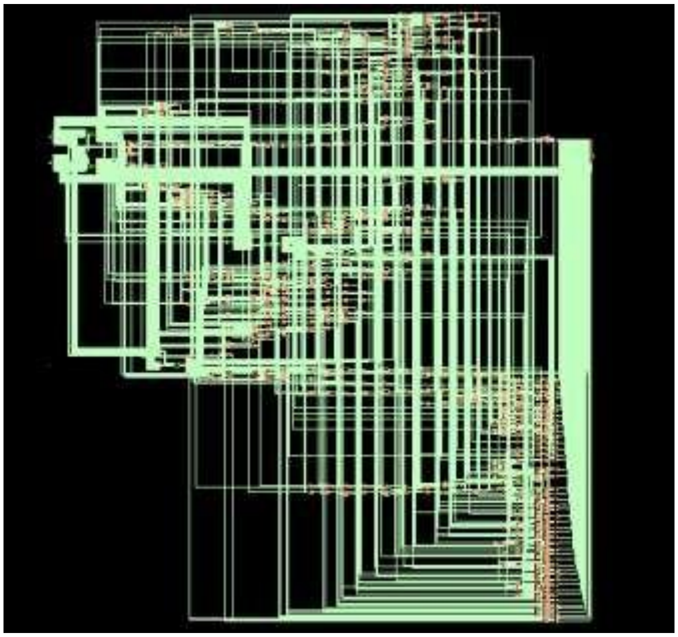
Step 4:- Innovus is a tool developed by Cadence Design Systems that is used for physical design and implementation of integrated circuits (ICs). The tool offers a wide range of features and functionalities that help design engineers to optimize their designs and achieve the best possible performance with minimal power consumption.

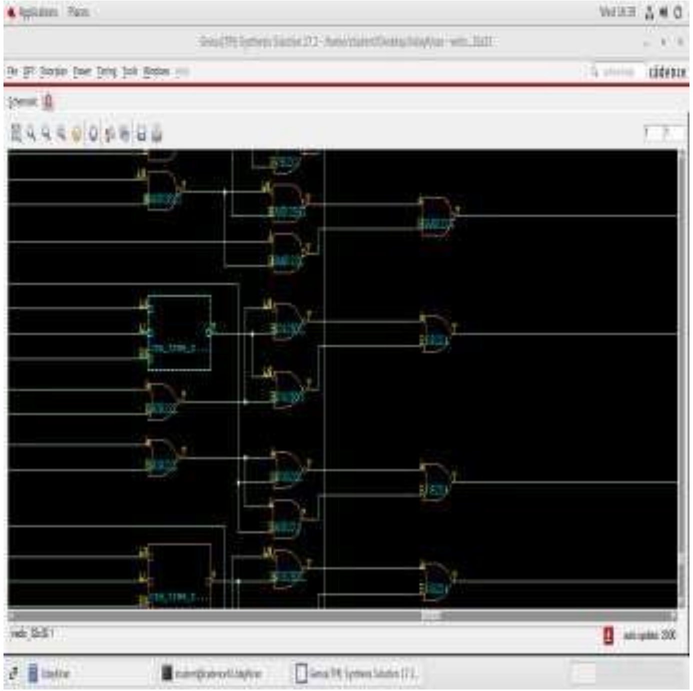
6. RESULTS

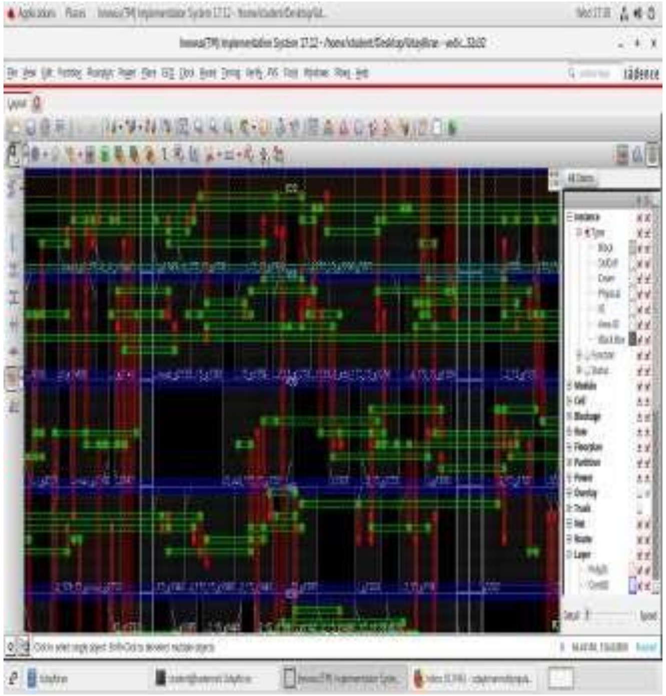
6.1 SIMULATION OF 32-BIT VEDIC MULTIPLIER USING NC LAUNCH



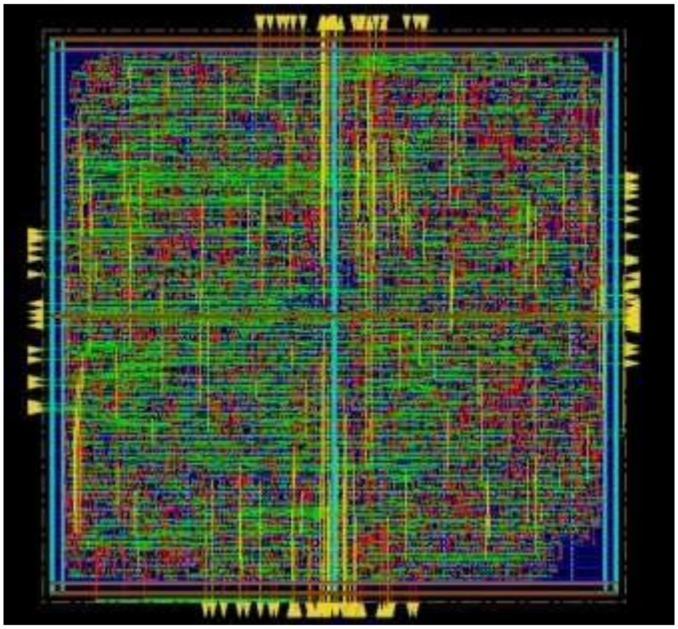
6.2 SYNTHESIS OF 32-BIT VEDIC MULTIPLIER USING GENUS



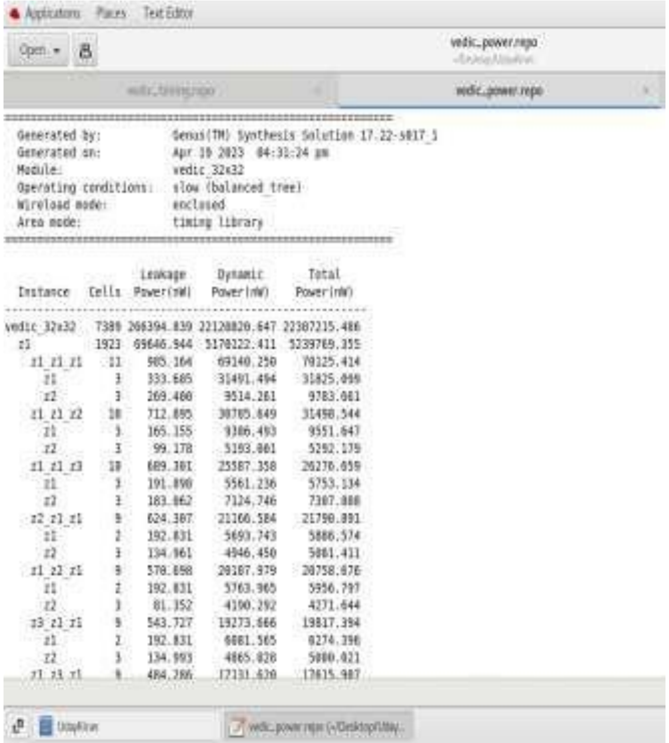




6.3 LAYOUT OF 32-BIT VEDIC MULTIPLIER USING INNOVUS

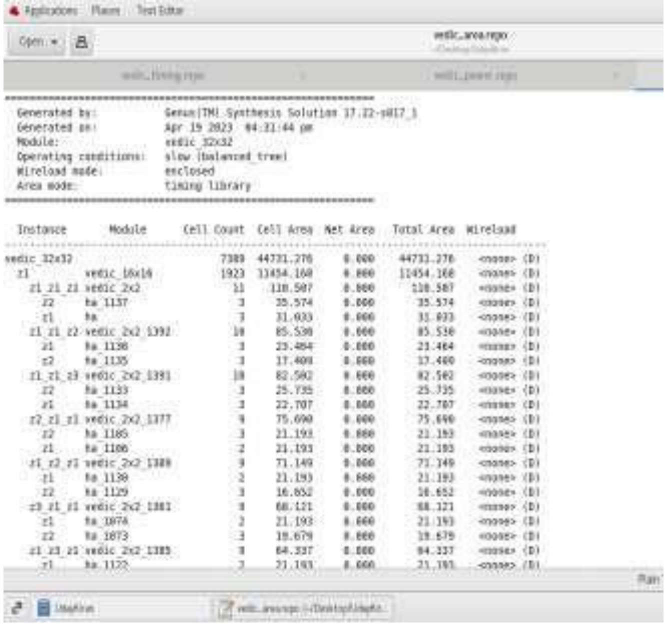


6.4 POWER REPORT OF 32-BIT VEDIC MULTIPLIER



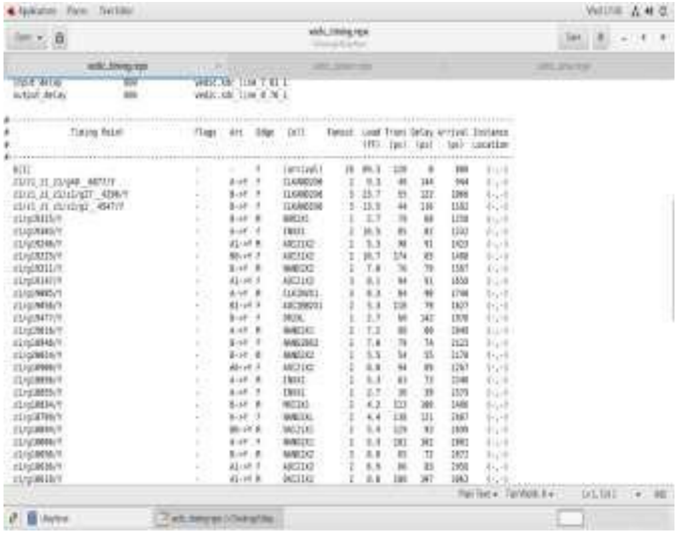
```
================================================================
  Generated by:            Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:            Apr 19 2023  04:31:24 pm
  Module:                  vedic_32x32
  Operating conditions:    slow (balanced_tree)
  Wireload mode:           enclosed
  Area mode:               timing library
================================================================

                        Leakage      Dynamic        Total
  Instance     Cells   Power(nW)    Power(nW)     Power(nW)
-----------------------------------------------------------------
vedic_32x32    7389  266394.839 22120820.647 22387215.486
  z1           1923   69646.944  5170122.411  5239769.355
   z1_z1_z1      11     985.164    69140.250    70125.414
     z1           3     333.605    31491.494    31825.099
     z2           3     269.400     9514.261     9783.661
   z1_z1_z2      10     712.895    30785.649    31498.544
     z1           3     165.155     9386.493     9551.647
     z2           3      99.178     5193.001     5292.179
   z1_z1_z3      10     689.301    25587.358    26276.659
     z1           3     191.898     5561.236     5753.134
     z2           3     183.062     7124.746     7307.808
   z2_z1_z1       9     624.307    21166.584    21790.891
     z1           2     192.831     5693.743     5886.574
     z2           3     134.961     4946.450     5081.411
   z1_z2_z1       9     570.898    20187.979    20758.876
     z1           2     192.831     5763.965     5956.797
     z2           3      81.352     4190.292     4271.644
   z3_z1_z1       9     543.727    19273.666    19817.394
     z1           2     192.831     6081.565     6274.396
     z2           3     134.993     4865.028     5000.021
   z1_z3_z1       9     484.286    17131.620    17615.907
```

6.5 AREA REPORT OF 32-BIT VEDIC MULTIPLIER

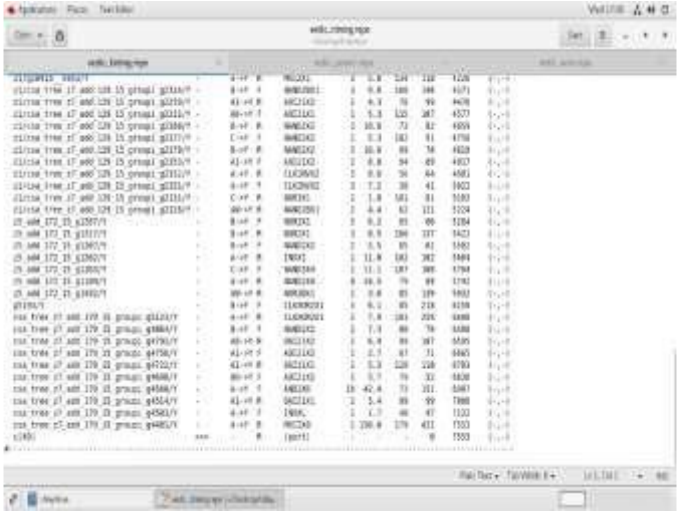

Applications Places Text Editor
Open
vedic_area.repo
vedic_timing.repo
vedic_power.repo
Plain
Utkarsh
vedic_area.repo (~/Desktop/Utkarsh)

```
=========================================================================
  Generated by:           Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:           Apr 19 2023  04:31:44 pm
  Module:                 vedic_32x32
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
=========================================================================

  Instance     Module      Cell Count  Cell Area  Net Area   Total Area  Wireload
--------------------------------------------------------------------------------
vedic_32x32                      7389  44731.276     0.000    44731.276   <none> (D)
  z1          vedic_16x16        1923  11454.160     0.000    11454.160   <none> (D)
    z1_z1_z1  vedic_2x2            11    110.507     0.000      110.507   <none> (D)
      z2      ha_1137               3     35.574     0.000       35.574   <none> (D)
      z1      ha                    3     31.033     0.000       31.033   <none> (D)
    z1_z1_z2  vedic_2x2_1392       10     85.530     0.000       85.530   <none> (D)
      z1      ha_1136               3     23.464     0.000       23.464   <none> (D)
      z2      ha_1135               3     17.400     0.000       17.400   <none> (D)
    z1_z1_z3  vedic_2x2_1391       10     82.502     0.000       82.502   <none> (D)
      z2      ha_1133               3     25.735     0.000       25.735   <none> (D)
      z1      ha_1134               3     22.707     0.000       22.707   <none> (D)
    z2_z1_z1  vedic_2x2_1377        9     75.690     0.000       75.690   <none> (D)
      z2      ha_1105               3     21.193     0.000       21.193   <none> (D)
      z1      ha_1106               2     21.193     0.000       21.193   <none> (D)
    z1_z2_z1  vedic_2x2_1389        9     71.149     0.000       71.149   <none> (D)
      z1      ha_1130               2     21.193     0.000       21.193   <none> (D)
      z2      ha_1129               3     16.652     0.000       16.652   <none> (D)
    z3_z1_z1  vedic_2x2_1381        9     68.121     0.000       68.121   <none> (D)
      z1      ha_1074               2     21.193     0.000       21.193   <none> (D)
      z2      ha_1073               3     19.679     0.000       19.679   <none> (D)
    z1_z3_z1  vedic_2x2_1385        8     64.337     0.000       64.337   <none> (D)
      z1      ha_1122               2     21.193     0.000       21.193   <none> (D)
```

6.6 TIMING REPORT OF 32-BIT VEDIC MULTIPLIER



```
  input_delay            800          vedic.sdc_line_7_61_1
  output_delay           800          vedic.sdc_line_8_76_1

#------------------------------------------------------------------------------------------
#          Timing Point               Flags  Arc  Edge  Cell      Fanout  Load Trans Delay Arrival Instance
#                                                                          (fF)  (ps)  (ps)   (ps)  Location
#------------------------------------------------------------------------------------------
  [illegible]
```

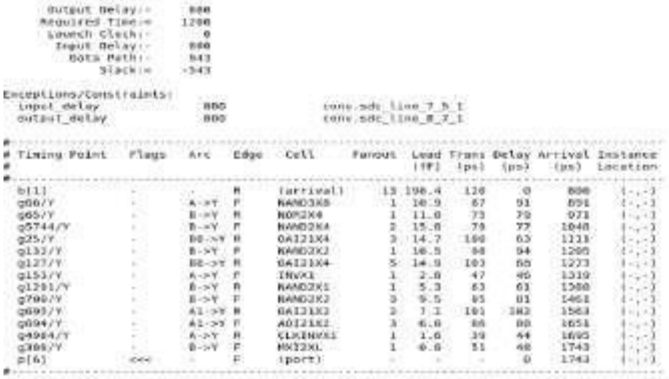


```
  [illegible]
  c[63]                              <<<        F     (port)                     0    7553
#------------------------------------------------------------------------------------------
```

7. PERFORMANCE COMPARISIONS OF4BITCONVENTIONAL AND VEDIC MULTIPLIERS.

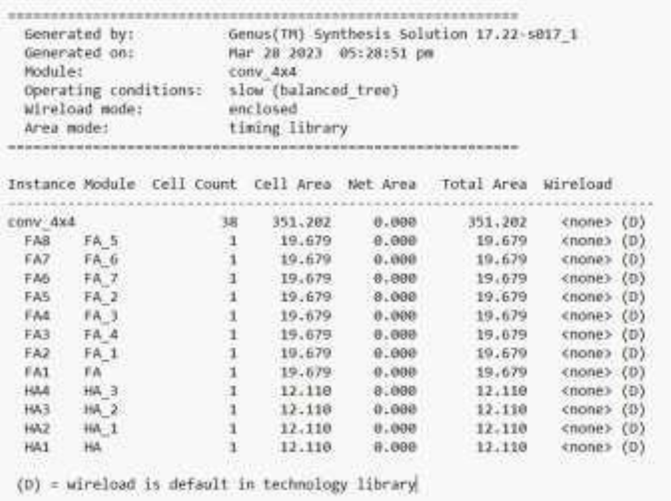
7.1CONVENTIONAL 4 BIT MULTIPLIER

7.1.1 TIMING REPORT OF 4BIT CONVENTIONAL MULTIPLIER



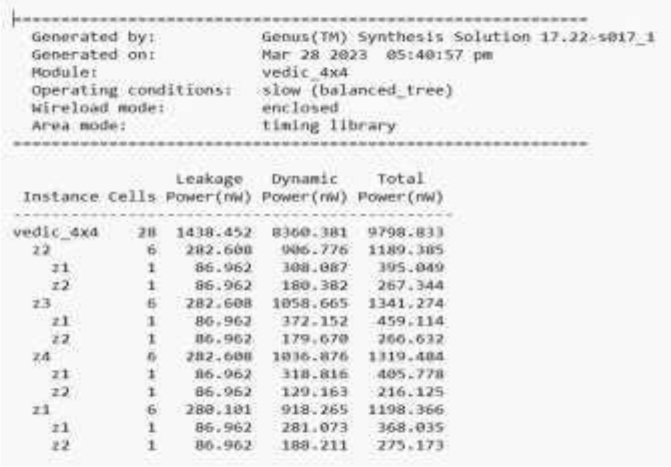
```
      Output Delay:-     800
     Required Time:=    1200
      Launch Clock:-       0
       Input Delay:-     800
         Data Path:-     943
             Slack:=    -543

Exceptions/Constraints:
  input_delay             800              conv.sdc_line_7_5_1
  output_delay            800              conv.sdc_line_8_7_1

#---------------------------------------------------------------------------------------------------
# Timing Point   Flags   Arc    Edge   Cell         Fanout   Load  Trans  Delay  Arrival  Instance
#                                                            (fF)   (ps)   (ps)     (ps)  Location
#---------------------------------------------------------------------------------------------------
  b[1]             -      -       R     (arrival)       13  196.4    126      0      800    (-,-)
  g66/Y            -     A->Y     F     NAND3X8          1   10.9     67     91      891    (-,-)
  g65/Y            -     B->Y     R     NOR2X4           1   11.0     73     79      971    (-,-)
  g5744/Y          -     B->Y     F     NAND2X4          2   15.8     78     77     1048    (-,-)
  g25/Y            -     B0->Y    R     OAI21X4          3   14.7    100     63     1111    (-,-)
  g133/Y           -     B->Y     F     NAND2X2          1   10.5     88     94     1205    (-,-)
  g127/Y           -     B0->Y    R     OAI21X4          5   14.9    103     68     1273    (-,-)
  g153/Y           -     A->Y     F     INVX1            1    2.8     47     46     1319    (-,-)
  g1281/Y          -     B->Y     R     NAND2X1          1    5.3     63     61     1380    (-,-)
  g700/Y           -     B->Y     F     NAND2X2          3    9.5     85     81     1461    (-,-)
  g693/Y           -     A1->Y    R     OAI31X2          2    7.1    101    102     1563    (-,-)
  g694/Y           -     A1->Y    F     AOI21X2          3    6.8     88     88     1651    (-,-)
  g4984/Y          -     A->Y     R     CLKINVX1         1    1.6     39     44     1695    (-,-)
  g306/Y           -     B->Y     F     MXI2XL           1    0.8     51     48     1743    (-,-)
  p[6]            <<<     -       F     (port)           -      -      -      0     1743    (-,-)
#---------------------------------------------------------------------------------------------------
```

7.1.2 POWER REPORT OF 4 BIT CONVENTIONAL MULTIPLIER



```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:           Mar 28 2023  05:28:51 pm
  Module:                 conv_4x4
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

Instance Module  Cell Count  Cell Area  Net Area   Total Area  Wireload
-----------------------------------------------------------------------
conv_4x4               38     351.202     0.000     351.202    <none> (D)
  FA8    FA_5           1      19.679     0.000      19.679    <none> (D)
  FA7    FA_6           1      19.679     0.000      19.679    <none> (D)
  FA6    FA_7           1      19.679     0.000      19.679    <none> (D)
  FA5    FA_2           1      19.679     0.000      19.679    <none> (D)
  FA4    FA_3           1      19.679     0.000      19.679    <none> (D)
  FA3    FA_4           1      19.679     0.000      19.679    <none> (D)
  FA2    FA_1           1      19.679     0.000      19.679    <none> (D)
  FA1    FA             1      19.679     0.000      19.679    <none> (D)
  HA4    HA_3           1      12.110     0.000      12.110    <none> (D)
  HA3    HA_2           1      12.110     0.000      12.110    <none> (D)
  HA2    HA_1           1      12.110     0.000      12.110    <none> (D)
  HA1    HA             1      12.110     0.000      12.110    <none> (D)

 (D) = wireload is default in technology library
```

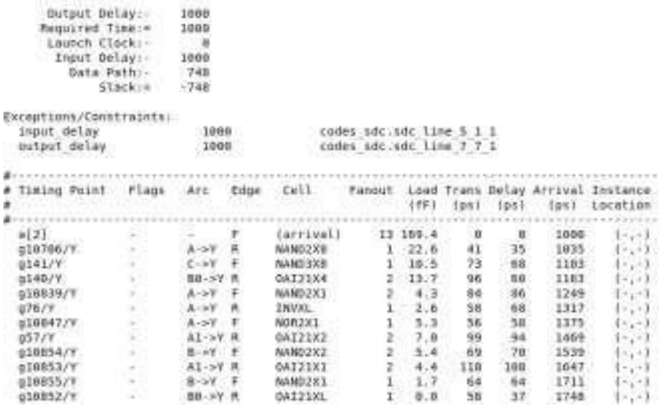
7.1.3 AREA REPORT OF 4 BIT CONVENTIONAL MULTIPLIER



```
============================================================
  Generated by:           Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:           Mar 28 2023  05:40:57 pm
  Module:                 vedic_4x4
  Operating conditions:   slow (balanced_tree)
  Wireload mode:          enclosed
  Area mode:              timing library
============================================================

                    Leakage   Dynamic    Total
 Instance Cells Power(nW) Power(nW) Power(nW)
-----------------------------------------------
vedic_4x4     28  1438.452  8360.381  9798.833
  z2           6   282.608   906.776  1189.385
    z1         1    86.962   308.087   395.049
    z2         1    86.962   180.382   267.344
  z3           6   282.608  1058.665  1341.274
    z1         1    86.962   372.152   459.114
    z2         1    86.962   179.670   266.632
  z4           6   282.608  1036.876  1319.484
    z1         1    86.962   318.816   405.778
    z2         1    86.962   129.163   216.125
  z1           6   280.101   918.265  1198.366
    z1         1    86.962   281.073   368.035
    z2         1    86.962   188.211   275.173
```

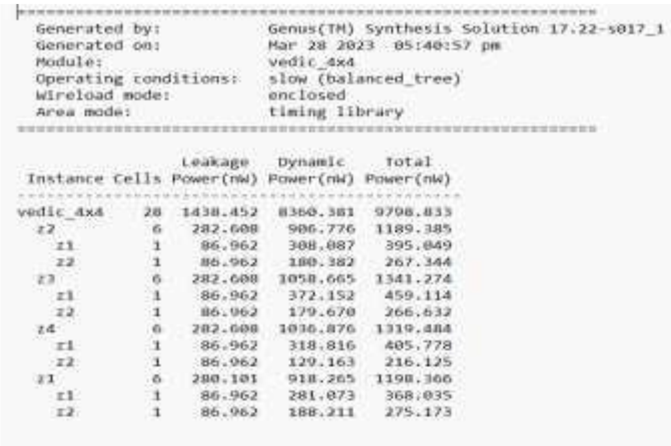
7.2 VEDIC 4 BIT MULTIPLIER

7.2.1 TIMING REPORT OF 4BIT VEDIC MULTIPLIER



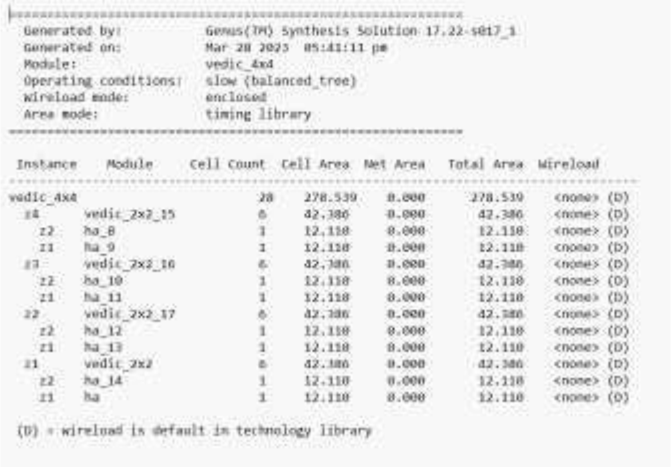
```
       Output Delay:-    1000
     Required Time:=     1000
      Launch Clock:-        0
       Input Delay:-     1000
         Data Path:-      748
             Slack:=     -748

Exceptions/Constraints:
  input_delay              1000            codes_sdc.sdc_line_5_1_1
  output_delay             1000            codes_sdc.sdc_line_7_7_1

#--------------------------------------------------------------------------------------
# Timing Point   Flags   Arc    Edge   Cell        Fanout  Load Trans Delay Arrival Instance
#                                                          (fF)  (ps)  (ps)   (ps)  Location
#--------------------------------------------------------------------------------------
  a[2]           -       -      F      (arrival)       13 189.4     0     0   1000    (-,-)
  g10706/Y       -       A->Y   R      NAND2X0          1  22.6    41    35   1035    (-,-)
  g141/Y         -       C->Y   F      NAND3X8          1  10.5    73    68   1103    (-,-)
  g140/Y         -       B0->Y  R      OAI31X4          2  13.7    96    80   1183    (-,-)
  g10839/Y       -       A->Y   F      NAND2X1          2   4.3    86    66   1249    (-,-)
  g76/Y          -       A->Y   R      INVXL            1   2.6    58    68   1317    (-,-)
  g10847/Y       -       A->Y   F      NOR2X1           1   5.3    56    58   1375    (-,-)
  g57/Y          -       A1->Y  R      OAI21X2          2   7.0    99    94   1469    (-,-)
  g10854/Y       -       B->Y   F      NAND2X2          2   5.4    69    70   1539    (-,-)
  g10853/Y       -       A1->Y  R      OAI21X1          2   4.4   110   108   1647    (-,-)
  g10855/Y       -       B->Y   F      NAND2X1          1   1.7    64    64   1711    (-,-)
  g10852/Y       -       B0->Y  R      OAI21XL          1   0.0    58    37   1748    (-,-)
```

7.2.2 POWER REPORT OF 4BIT VEDIC MULTIPLIER



```
+============================================================
  Generated by:            Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:            Mar 28 2023  05:40:57 pm
  Module:                  vedic_4x4
  Operating conditions:    slow (balanced_tree)
  Wireload mode:           enclosed
  Area mode:               timing library
============================================================

                   Leakage    Dynamic     Total
 Instance Cells Power(nW) Power(nW) Power(nW)
-------------------------------------------------
vedic_4x4     28  1438.452   8360.381  9798.833
  z2           6   282.608    906.776  1189.385
    z1         1    86.962    308.087   395.049
    z2         1    86.962    180.382   267.344
  z3           6   282.608   1058.665  1341.274
    z1         1    86.962    372.152   459.114
    z2         1    86.962    179.670   266.632
  z4           6   282.608   1036.876  1319.484
    z1         1    86.962    318.816   405.778
    z2         1    86.962    129.163   216.125
  z1           6   280.101    918.265  1198.366
    z1         1    86.962    281.073   368.035
    z2         1    86.962    188.211   275.173
```

7.2.3 AREA REPORT OF 4BIT VEDIC MULTIPLIER



```
+============================================================
  Generated by:            Genus(TM) Synthesis Solution 17.22-s017_1
  Generated on:            Mar 28 2023  05:41:11 pm
  Module:                  vedic_4x4
  Operating conditions:    slow (balanced_tree)
  Wireload mode:           enclosed
  Area mode:               timing library
============================================================

 Instance   Module        Cell Count  Cell Area  Net Area   Total Area  Wireload
----------------------------------------------------------------------------------
vedic_4x4                         28    278.539     0.000      278.539   <none> (D)
  z4      vedic_2x2_15             6     42.386     0.000       42.386   <none> (D)
    z2    ha_8                     1     12.110     0.000       12.110   <none> (D)
    z1    ha_9                     1     12.110     0.000       12.110   <none> (D)
  z3      vedic_2x2_16             6     42.386     0.000       42.386   <none> (D)
    z2    ha_10                    1     12.110     0.000       12.110   <none> (D)
    z1    ha_11                    1     12.110     0.000       12.110   <none> (D)
  z2      vedic_2x2_17             6     42.386     0.000       42.386   <none> (D)
    z2    ha_12                    1     12.110     0.000       12.110   <none> (D)
    z1    ha_13                    1     12.110     0.000       12.110   <none> (D)
  z1      vedic_2x2                6     42.386     0.000       42.386   <none> (D)
    z2    ha_14                    1     12.110     0.000       12.110   <none> (D)
    z1    ha                       1     12.110     0.000       12.110   <none> (D)

(D) = wireload is default in technology library
```

8 CONCLUSIONS

The 32-bit Vedic multiplier is a powerful tool for performing multiplication operations in digital circuits. It utilizes Vedic mathematics principles to provide high-speed and accurate multiplication results with low power consumption. However, there are some limitations to its use, such as being limited to multiplication operations, requiring specialized knowledge for implementation, and having more complex implementation than traditional multiplication methods.

Despite these drawbacks, the 32-bit Vedic multiplier has a number of benefits over conventional multiplication techniques. It uses less power, is easily implemented in hardware or software, and is extremely scalable. Because of this, it is perfect for use in systems that demand high-speed multiplication, such as digital signal processing systems, microprocessors, and other digital circuits.

Overall, the 32-bit Vedic multiplier is a valuable tool for digital circuit design, and its advantages and disadvantages should be carefully considered before implementing it in a particular application. Its ability to provide fast and accurate multiplication results with low power consumption makes it an attractive option for many digital circuit designs.

Multiplier	Time Delay (ns)	Power Consumption (µs)	Area Taken
Conventional 4 bit	0.943	11.488	351.202
Vedic 4 bit	0.748	9.798	278.53

9. REFERENCES

8.1.1.1 Jagadguru Swami Sri Bharati Krishna Tirthaji Maharaja, "Vedic Mathematics: Sixteen Simple Mathematical Formulae from the Veda" pp. 5- 45, Motilal Banarasidas Publishers, Delhi, 2009.

8.1.1.2 Sai Venkatramana Prasada G S, G Seshikala, Niranjana S "Design of High Speed 32-bit Floating Point Multiplier using Urdhva Tiryagbhyam Sutra of Vedic Mathematics", International Journal of Recent Technology and Engineering (IJRTE), Vol. 8, Issue- 2S3, July 2019.

8.1.1.3 Manpreet Kaur, Amandeep Kaur, "Design and Analysis of Vedic Multiplier by Using Modified Full Adders", International Journal of Engineering Development and Research (IJEDR), Vol. 6, Issue 2, 2018.

8.1.1.3.1 P. Manju, P.B. Sreelakshmi, T. Madhavi, "Design of Vedic Multiplier Using GDI method", International Journal of Recent Technology and Engineering (IJRTE), Vol. 7, Issue 6S4, April 2019.

8.1.1.3.2 Arunkumar P. Chavan, Rahul Verma, Nishanth S. Bhat “High Speed 32-bit Vedic Multiplier for DSP Applications”, International Journal of Computer Applications, Vol.135- No.7, February 2016.

8.1.1.3.3 Siba Kumar Panda, Ritisnigdha Das, S K Saifur Raheman, Tapasa Ranjan Sahoo, “VLSI Implementation of Vedic Multiplier Using Urdhva Tiryagbhyam Sutra in VHDL Environment: A Novelty”, IOSR Journal of VLSI and Signal Processing (IOSR-JVSP), Vol.5, Issue 1, pp 17-24, Jan-Feb 2015.

8.1.1.3.4 Syed Shahzad Hussain Shah, Muhammad Naseem Majoka, Gulistan Raja, “Design and Implementation of 32-bit Vedic Multiplier on FPGA”, First International Conference on Modern communication & Computing Technologies, Research Gate, February2014.

8.1.1.3.5 S. Nithya Devi, B. Gopinath, P. Ramanathan, “Synthesis of High Speed Vedic Multiplier”, International Journal of Applied Engineering Research, Vol.10, No. 29, May 2015.

8.1.1.3.6 S. Jayakumar, R. Selvam, K.Karthikeyan, R. Natesan, “Design and Implementation Hybrid FIR Filters using Vedic Multiplier and Fast Adders”, International Journal of Recent Technology and Engineering (IJRTE), Vol.8, Issue 4, November 2019.