Design of 4x4 bit Vedic Multiplier using HSPICE and implementing Low Power Techniques

Aakash Mhankale
California State University Fullerton
Graduate Student
803000470
aakashmhankale@csu.fullerton.edu

Deeksha Sivakumer
California State University Fullerton
Graduate Student
893513473
deeksha@csu.fullerton.edu

Abstract—

The need of high speed multiplier is increasing as the need of high speed processors are increasing. A Multiplier is one of the key hardware blocks in most fast processing system which is not only a high delay block but also a major source of power dissipation. A conventional processor requires substantially more hardware resources and processing time in the multiplication operation, rather than addition and subtraction. We are implementing a high speed 4x4 bit Vedic Multiplier (VM) based on Vertically & Crosswise method of Vedic mathematics, a general multiplication formulae equally applicable to all cases of multiplication. It is based on generating all partial products and their sum in one step. The coding is done in HSPICE as well as simulation. The Power dissipation of 4x4 bit Vedic multiplier calculated as a baseline at 130nm technology node and compared with various low power techniques. We found that the in some techniques the delay is increasing but the power dissipation is reducing significantly. So we can use these circuits for power efficient devices.

Keywords—

Vedic Multiplication, 4 bit multiplier, Ripple Carry Adder, low power techniques, coarse grain power gating-Vdd/Vss, fine grain power gating-Vdd/Vss, Reduced input swing, voltage supply scaling

1.INTRODUCTION

Multipliers are extensively used in Microprocessors, DSP and Communication applications. For higher order multiplications, a huge number of adders are to be used to perform the partial product addition. The need of low power and high speed Multiplier is increasing as the need of high speed processors are increasing. The mathematical operations using, Vedic Method are very fast and requires less hardware, this can be used to improve the computational speed of processors. Further, will describe hardware architecture of 2x2 and 4x4 bits Vedic Multiplier (VM) based on Vedic multiplication. Section 3 illustrates the implementation of low power techniques and result of Vedic multiplier deck obtained while Section 4 comprises of Conclusion. Section 5 is the Acknowledgment whereas Section 6 comprises of References.

2. Work. Methodology and tools

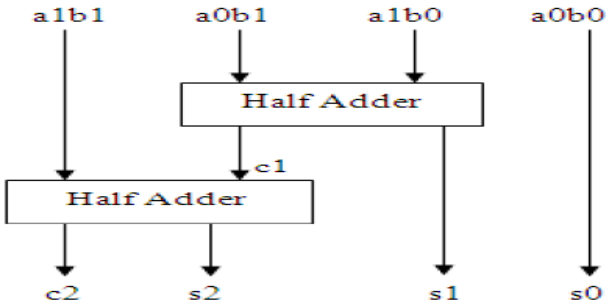
2.1 Vedic Multiplier for 2x2 bit

The method is explained below for two, 2 bit numbers A and B where A = a1a0 and B = b1b0 as shown in Figure 2. Firstly, the Least Significant Bits are multiplied which gives the Least Significant Bit (LSB) of the final product (vertical). Then, the LSB of the multiplicand is multiplied with the next higher bit of the multiplier and added with, the product of LSB of multiplier and next higher bit of the multiplicand (crosswise). The sum gives second bit of the final product and the carry is added with the partial product obtained by multiplying the most significant bits to give the sum and carry. The sum is the third corresponding bit and carry becomes the fourth bit of the final product.

s0 = a0b0

c1s1 = a1b0 + a0b1

c2s2 = c1 + a1b1



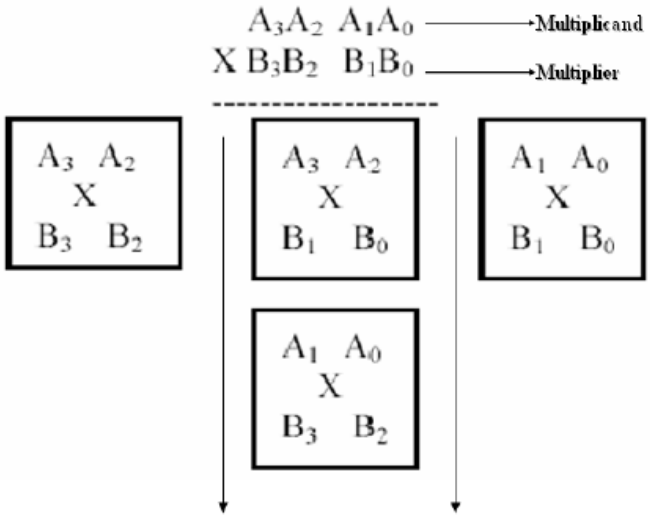
Block Diagram of 2x2 bit Vedic Multiplier (VM)

The final result will be c2s2s1s0. This multiplication method is applicable for all the cases. The 2x2 bit Vedic multiplier (VM) module is implemented using four input AND gates & two half-half adders.

The same method can be extended for higher no. of input bits (say 4). But a little modification is required. This section illustrates the implementation of 4x4 bit VM which uses 2x2 bit VM as a basic module.

2.2 Vedic Multiplier for 4x4 bit

Divide the no. of bits in the inputs equally in two parts. Let's analyse 4x4 bit multiplication, say multiplicand A=A3A2A1A0 and multiplier B= B3B2B1B0. Following are the output line for the multiplication result, S7S6S5S4S3S2S1S0. Let's divide A and B into two parts, say "A3 A2" & "A1 A0" for A and "B3 B2" & "B1B0" for B. Using the fundamental of Vedic multiplication, taking two bit at a time and using 2 bit multiplier block, we can have the following structure for 4x4 bit multiplication as shown below.



Structure for 4x4 bit Multiplication

Each block as shown above is 2x2 bit multiplier. First 2x2 multiplier inputs are "A1 A0" and "B1 B0". The last block is 2x2 bit multiplier with inputs "A3 A2" and "B3 B2". The middle one shows two, 2x2 bit multiplier with inputs "A3A2" & "B1B0" and "A1A0" & "B3B2". So the final result of multiplication, which is of 8 bit, "S7S6S5S4S3S2 S1S0".

To understand the concept, the block diagram of 4x4 bit Vedic multiplier is shown in Figure 1. To get final product S7S6S5S4S3S2S1S0 four, 2-bit Vedic multiplier and three 4-bit Ripple Carry (RC) Adders are required. We are using the first 4-bit RC Adder is used to add two 4-bit operands obtained from cross multiplication of the two middle 2x2 bit multiplier modules. The second 4-bit RC Adder is used to add two 4-bit operands, i.e. concatenated 4-bit ("00" & most significant two output bits of right hand most of 2x2 multiplier module as shown in Figure 1) and one 4-bit operand we get as the output sum of first RC Adder. Its carry "ca1" is forwarded to third RC Adder. Now the third 4-bit RC Adder is used to add two 4-bit operands, i.e. concatenated 4 -bit (carry ca1, "0" & most significant two output sum bits of 2nd RC Adder as shown in Figure 1) and one 4-bit operand we get as the output sum of left hand most of 2x2 multiplier module. The arrangement of Ripple Carry Adder as shown in Figure 2 helps us to reduce delay.

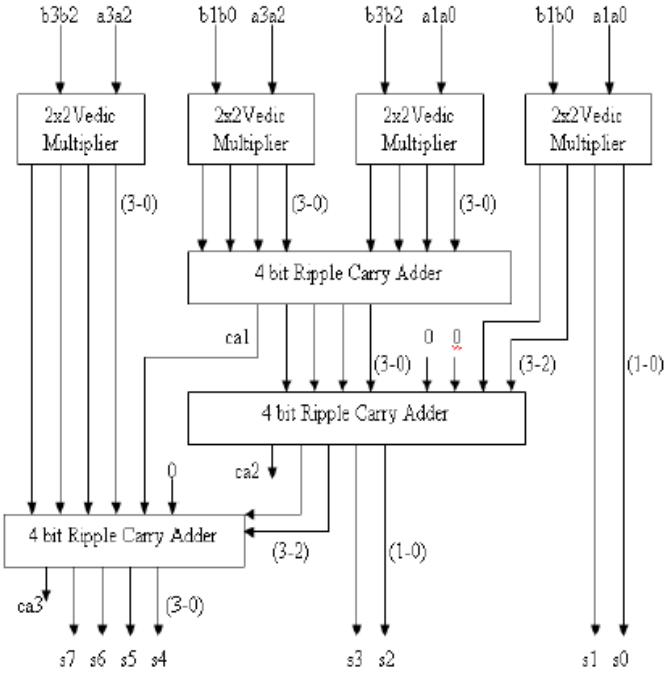


Figure 1 Block Diagram of 4x4 bit Vedic Multiplier (VM)

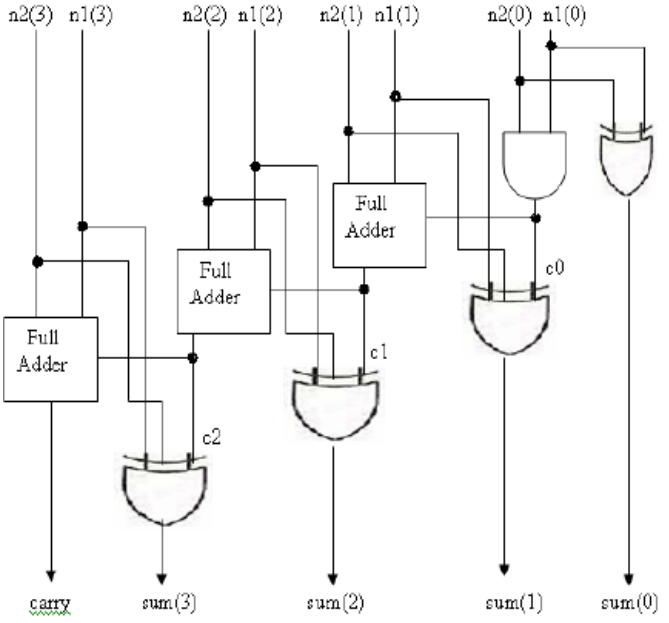


Figure 2 Circuit Diagram of 4 bit Ripple Carry Adder

3. *IMPLEMENTATION & RESULTS*

- **Baseline Result**

Avg. Leakage Power dissipation = 561 uW

	130nm Technology			
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(u A)	Static Power Dissipiation(u W)
1111	0000	1.3	127.0848	165.21024
0011	1111	1.3	196.3389	255.24057
1100	1111	1.3	196.3389	255.24057
1010	1111	1.3	139.0561	180.77293
0110	1001	1.3	358.5115	466.06495
		Total	1017.3302	1484.43728
Average Leakage Current			203.46604	
				2806.96654
Average Leakage Power				561.393308

Dynamic Power Dissipation : 1.4789 mW

Table 1 displays the comparison of simulation results of the proposed Vedic multiplier with the Conventional multipliers in terms of time delay (in nanoseconds). The combinational path delay obtained for the proposed 4x4 bit Vedic multiplier is less whereas the results of 4x4 bit Array and Booth multipliers have been taken from Umesh Akare [16].

Table 1 Comparison of Multipliers (in nanosecond)

Device: Spartan xc3s50a-5tq144	Array Multiplier	Booth Multiplier	Vedic Multiplier
4x4 bit VM	32.001 ns	16.276 ns	13.102 ns

- **Low power Techniques and Results**

1. Voltage supply Scaling

This is one of the best technique to reduce the power dissipation. P total= p dynamic+ p static there is Switching in CMOS due to change in input from 0 to 1 and respective output also changes so there is switching. The power dissipation can be reduced by changing the rising and fall time. This reduces gate leakage power and subthreshold leakage power dissipation, further it reduces dynamic leakage power as the quadratic dependence on voltage supply.

Technology Node(nm)	Dynamic Power Disspiation(u W)
130	1.0918
Delay(pS)	130nm Technology Node
Delay1	32.4002

- From above result we can see that there is minimization in dynamic power. Further, the leakage power reduction is shown in following table.

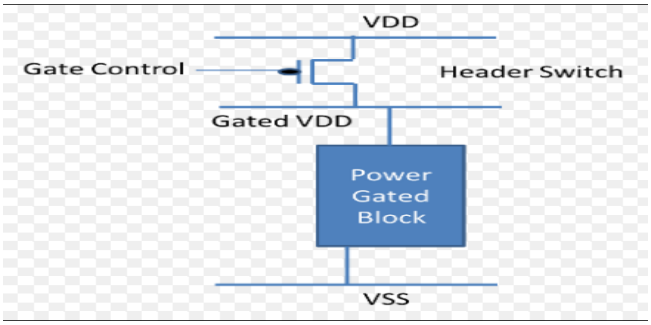
	130nm Technology			
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(u A)	Static Power Dissipiation(u W)
1111	0000	1.2	87.6647	171.99208
0011	1111	1.2	109.7967	202.30856
1100	1111	1.2	141.1806	158.78856
1010	1111	1.2	99.1981	189.23576
0110	1001	1.2	259.5068	183.92272
		Total	697.3469	1484.43728
		Average Leakage Current	139.46938	
				2390.68496
		Average Leakage Power		478.136992

.

14 % reduction in static leakage power compare to baseline

2. Coarse Grain Power Gating- Vdd

The coarse-grained approach implements sleep transistors which drives cells locally through shared virtual power networks.



Results: 20% reduction in leakage power.

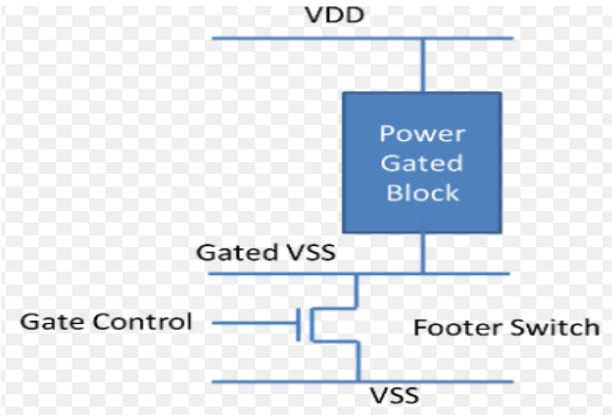
Significant reduction in dynamic power, but it is affecting the delay.

	130nm Technology			
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(u A)	Static Power Dissipiation(u W)
1111	0000	1.3	35.5482	46.21266
0011	1111	1.3	40.567	52.7371
1100	1111	1.3	38.678	50.2814
1010	1111	1.3	36.3259	47.22367
0110	1001	1.3	44.678	58.0814
		Total	195.7971	254.53623
		Average Leakage Current	39.15942	
				509.07246
		Average Leakage Power		101.814492

Dynamic Power Dissipation: 196.83 uW

Delay(pS)	130nm Technology Node
Delay1	39.7268

3. Coarse Grain Power Gating- Vss



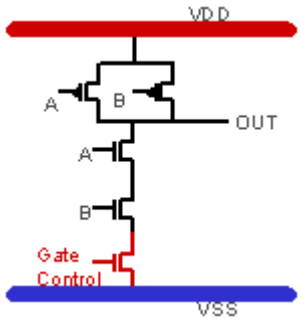
		130nm Technology		
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(n A)	Static Power Dissipiation(n W)
1111	0000	1.3	32.2604	41.93852
0011	1111	1.3	19.2725	25.05425
1100	1111	1.3	21.5125	27.96625
1010	1111	1.3	51.3982	66.81766
0110	1001	1.3	14.0892	18.31596
		Total	138.5328	180.09264
		Average Leakage Current	27.70656	
				360.18528
		Average Leakage Power		72.037056

Technology Node(nm)	Dynamic Power Disspiation(u W)
130	479.3507
Delay(pS)	130nm Technology Node
Delay1	29.5967

70% reduction in the leakage power dissipation.
50% reduction in the dynamic power dissipation.
Delay increase compared to baseline.

4. Fine grain power gating- vdd

Adding a sleep transistor to every cell that is to be turned off imposes a large area penalty, and individually gating the power of every cluster of cells creates timing issues. Fine-grain power gating encapsulates the switching transistor as a part of the standard cell logic.



Another issue is that the cells become more sensitive to PVT variations, because the built-in sleep transistor is subject to PVT variation which results in added IR-drop variation in the cell and hence performance variation.

		130nm Technology		
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(n A)	Static Power Dissipiation(n W)
1111	0000	1.3	60.3822	78.49686
0011	1111	1.3	47.6931	62.00103
1100	1111	1.3	28.7572	37.38436
1010	1111	1.3	48.3998	62.91974
0110	1001	1.3	37.9307	49.30991
		Total	223.163	290.1119
		Average Leakage Current	44.6326	
				580.2238
		Average Leakage Power		116.04476

Technology Node(nm)	Dynamic Power Disspiation(u W)
130	343.0589
Delay(pS)	130nm Technology Node
Delay1	37.63

68% decrease in Avg. leakage power dissipation.
46% decrease in Dynamic power dissipation.
But, the delay is increasing.

5. Fine grain power gating- vss

Results:

		130nm Technology		
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(u A)	Static Power Dissipiation(n W)
1111	0000	1.3	43.0745	55.99685
0011	1111	1.3	23.6259	30.71367
1100	1111	1.3	33.4152	43.43976
1010	1111	1.3	72.6302	94.41926
0110	1001	1.3	22.8043	29.64559
		Total	195.5501	254.21513
		Average Leakage Current	39.11002	
				508.43026
		Average Leakage Power		101.686052

Technology Node(nm)	Dynamic Power Disspiation(u W)
130	491.6856
Delay(pS)	130nm Technology Node
Delay1	28.4846

76 % reduction in leakage power reduction.
53% reduction in dynamic power reduction.

6. Reduced Input Swing

Results :

Technology Node(nm)	Dynamic Power Disspiation(m W)
130	1.6028
Delay(pS)	130nm Technology Node
Delay1	48.3071

	130nm Technology			
A	B			
A4A3A2A0	B4B23B2B1	Vdd	Leakage Current(u A)	Static Power Dissipiation(n W)
1111	0000	1.3	127.0848	165.21024
0011	1111	1.3	169.7251	220.64263
1100	1111	1.3	196.3389	255.24057
1010	1111	1.3	139.0561	180.77293
0110	1001	1.3	358.5115	466.06495
		Total	990.7164	1287.93132
		Average Leakage Current	198.14328	
				2575.86264
		Average Leakage Power		515.172528

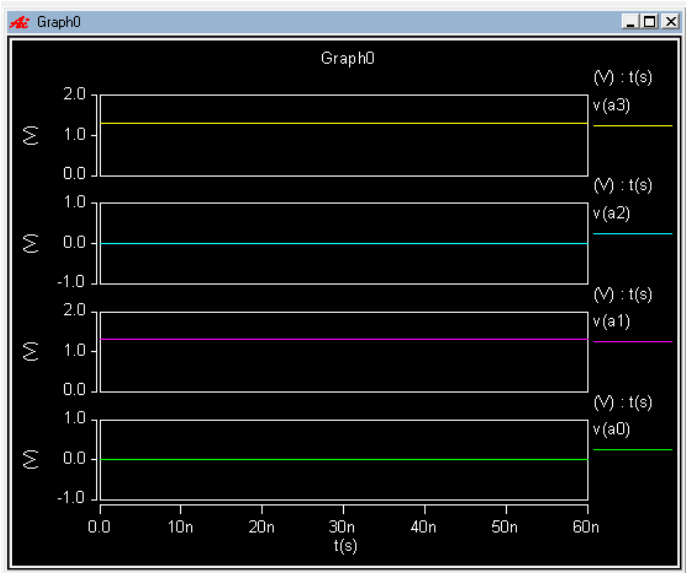
2% reduction in the leakage power dissipation.

1% reduction in dynamic power dissipation.

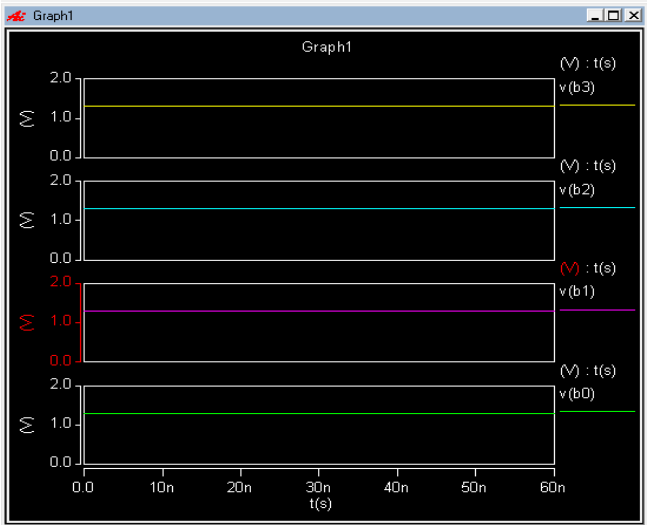
Further, significant rise in the delay which is not desirable.

- Screen Shots of static input combination

Input A 4 bit: A3A2A1A0 = “1010”

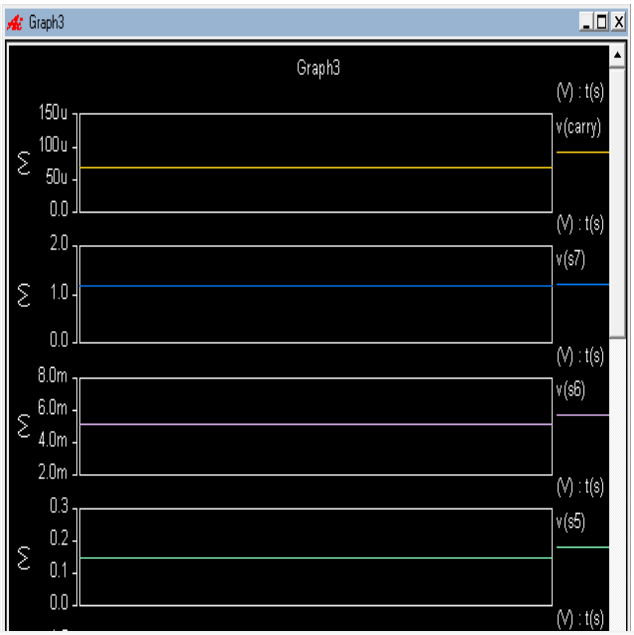


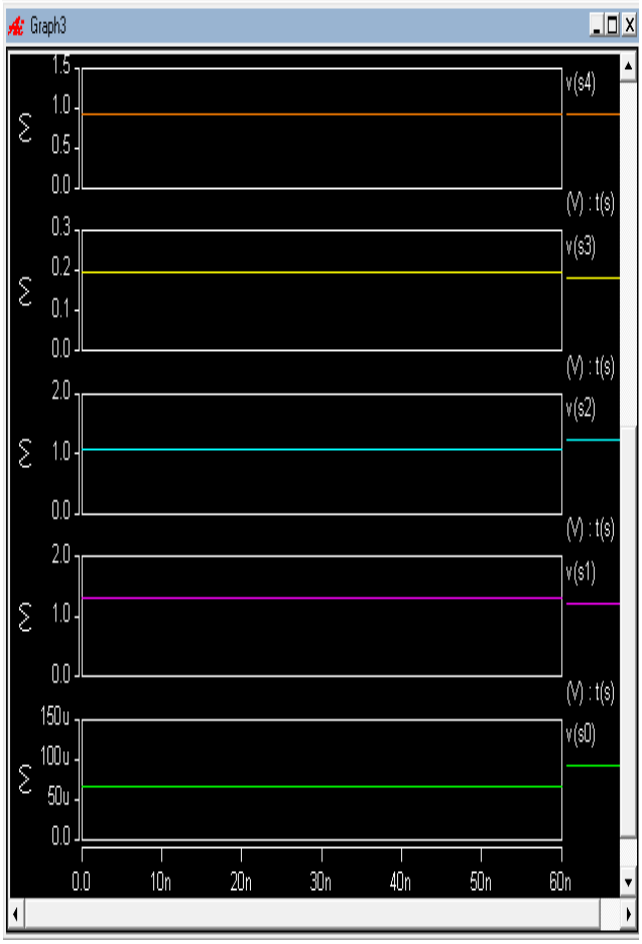
Input B 4 bit: B3B2B1B0 = “1111”



Output:

“Carry S7S6S5S4S3S2S1S0”= “0 10010110”





The above Cscope implementation shows the correctness of the circuits and giving desired results.

4 . Future Work

We can implement other power reduction techniques with the baseline such as Multi-threshold V-th, Sleepy stack, use of finFet transistors any many more. Further, we can implement using different technology nodes like 22nm, 32nm etc. In future we can implement circuit on FPGA boards and can get the real time statistic results which will give us more idea about the power and delay by synthesizing in Xilinx tool.

5. Conclusions

Baseline calculation done for the 4 bit vedic multiplier. In power scaling technique we can see reduction average leakage power and the dynamic power dissipation decrease and the delay in the circuit increased. In coarse grain gated with VDD, the dynamic power reduced as well as the leakage power also reduced but the delay increased. Using coarse gain with VSS dynamic power reduced more than dynamic power of coarse gain with VDD technique, while the leakage current also decreased significantly. In fine grain with VDD dynamic power dissipation reduced more as compared to coarse gain with VDD, and its leakage power increased (three times) as we changed the technique. For fine grain with VSS the leakage power dissipation was decreased as compared to fine grain with VDD

• Techniques must be devised to reduce power dissipation
• Techniques must be devised to accurately estimate the power dissipation
• There is advantage as well as trade of is also there, so to get correct simulation result we have to do more experiments with mentioned low power techniques.

References

[1] Jagadguru Swami Sri Bharati Krisna Tirthaji Maharaja, "Vedic Mathematics or Sixteen Simple Mathematical Formulae from the Veda, Delhi (1965)", Motilal Banarsidas, Varanasi, India, 1986.

[2] M. Morris Mano, "Computer System Architecture", 3rd edition, Prientice-Hall, New Jersey, USA, 1993, pp. 346-348.

[3] H. Thapliyal and H.R Arbania. "A Time-Area-Power Efficient Multiplier and Square Architecture Based On Ancient Indian Vedic Mathematics", Proceedings of the 2004, International Conference on VLSI (VLSI'04), Las Vegas, Nevada, June 2004, pp. 434-9.

[4] P. D. Chidgupkar and M. T. Karad, "The Implementation of Vedic Algorithms in Digital Signal Processing", Global J. of Engg. Edu, Vol.8, No.2, 2004 UICEE Published in Australia.

[5] Thapliyal H. and Srinivas M.B, "High Speed Efficient NxN Bit Parallel Hierarchical Overlay Multiplier Architecture Based on Ancient Indian Vedic Mathematics", Transactions on Engineering, Computing and Technology, 2004, Vol.2.

[6] Shamim Akhter, "VHDL Implementation of Fast NXN Multiplier Based On Vedic Mathematics". Jaypee Institute of Information Technology University, Noida, 201307 UP, INDIA, 2007 IEEE.

[7] Harpreet Singh Dhillon and Abhijit Mitra, "A Reduced– Bit Multipliction Algorithm for Digital Arithmatics", International Journal of Computational and Mathematical Sciences 2.2 @ www.waset.orgSpring2008.

[8] Honey Durga Tiwari, Ganzorig Gankhuyag, Chan Mo Kim, Yong Beom Cho, "Multiplier design based on ancient Indian Vedic Mathematician", International SoC Design Conference, 2008.

[9] Parth Mehta, Dhanashri Gawali, "Conventional versus Vedic mathematics method for Hardware implementation of a multiplier", International conference on Advance in Computing, Control, and Telecommunication Technologies, 2009.

[10] Ramalatha, M.Dayalan, K D Dharani, P Priya, and S Deoborah, "High Speed Energy Efficient ALU design using Vedic multiplication techniques", International Conference on Advances in Computational Tools for Engineering Applications, 2009. ACTEA '09.pp. 600-3, Jul 15-17, 2009.

[11] Sumita Vaidya and Deepak Dandekar, "Delay-Power Performance comparison of Multipliers in VLSI Circuit Design", International Journal of Computer Networks & Communications (IJCNC), Vol.2, No.4, July 2010.

[12] S.S.Kerur, Prakash Narchi, Jayashree C N, Harish M Kittur and Girish V A "Implementation of Vedic Multiplier For Digital Signal Processing" International conference on VLSI communication & instrumentation (ICVCI) 2011.

[13] Asmita Haveliya, "A Novel Design for High Speed Multiplier for Digital Signal Processing Applications (Ancient Indian Vedic mathematics approach)",

International Journal of Technology and Engineering System (IJTES), Vol.2, No.1, Jan-March, 2011.

[14] Prabha S., Kasliwal, B.P. Patil and D.K. Gautam, "Performance Evaluation of Squaring Operation by Vedic Mathematics", IETE Journal of Research, vol.57, Issue 1, Jan-Feb 2011.

[15] Aniruddha Kanhe, Shishir Kumar Das and Ankit Kumar Singh, "Design and Implementation of Low Power Multiplier Using Vedic Multiplication Technique", (IJCSC) International Journal of Computer Science and Communication Vol. 3, No. 1, January-June 2012, pp. 131-132

[16] Umesh Akare, T.V. More and R.S. Lonkar, "Performance Evaluation and Synthesis of Vedic Multiplier", National Conference on Innovative Paradigms in Engineering & Technology (NCIPET-2012), proceedings published by International Journal of Computer Applications (IJCA), 2012.

- **Work Contribution**

Deeksha Sivakumer (893513473)

1. Implementation of Half Adder circuit
2. Implementation of 2*2 bit multiplier
3. Creating sub circuit of 2*2 bit multiplier
4. Testing desired outputs with Cscope
5. Simulating the static and dynamic calculations in Hspise
6. Implementation of voltage supply scaling and calculation of Static, dynamic in Hspice
7. Implementation of reduced input swing and calculation of Static, dynamic in Hspice
8. Implementation of Coarse grain power gating-vss and calculation of Static, dynamic in Hspice
9. Respective Project report editing

Aakash Mhankale (803000470)

1. Implementation of Ripple carry Adder circuit
2. Implementation of 4*4 bit multiplier
3. Creating sub circuit for 4*4 bit multiplier
4. Testing desired outputs with Cscope
5. Simulating the static and dynamic calculations in Hspise
6. Implementation of Coarse grain power gating-vdd and calculation of Static, dynamic in Hspice
7. Implementation of Fine grain power gating-vss and calculation of Static, dynamic in Hspice
8. Implementation of Fine grain power gating-vdd and calculation of Static, dynamic in Hspice
9. Respective Project report editing