<u>10th August 2014. Vol. 66 No.1</u>

© 2005 - 2014 JATIT & LLS. All rights reserved

ISSN: 1992-8645

www.jatit.org

DESIGN OF LOW POWER MULTIPLIER USING COMPOUND CONSTANT DELAY LOGIC STYLE

¹ S. DARWIN, ² A. BENO, ³ L. VIJAYA LAKSHMI

 ^{1 & 2} Assistant Professor – Electronics & Communication Engineering Department, Dr. Sivanthi Aditanar College of Engineering, Tamilnadu, INDIA
 ³ PG Scholar, ME VLSI Design, Dr.Sivanthi Aditanar College of Engineering, Tamilnadu, INDIA E-mail: ¹ssdarwin@live.com, ²benoaustin@gmail.com ³vijeyalaxmi@gmail.com

ABSTRACT

High performance, energy efficient logic style is a popular research topic in the field of very large scale integrated (VLSI) circuits. A complex constant logic style is used to implement a logic expression to achieve high speed operation. This logic style is well suited for arithmetic circuit where critical path comprises of large cascaded inverting gates. Multiplication is a most utilized arithmetic operator that forms a part of filters, convolvers, and transforms processors in digital signal processing applications. This paper focuses on the design of the Wallace tree multiplier, Baugh wooley and Array multiplier using static logic style, dynamic logic style and compound constant delay logic style. The performance of energy delay product of Wallace tree multiplier, array multiplier and Baugh wooley multiplier using compound constant delay logic style is reduced considerably while compared to static and dynamic logic style.

Keywords: Wallace Tree multiplier, Array multiplier, Carry Skip adder, Cadence and Baugh Wooley Multiplier

1. INTRODUCTION

In low power Very Large Scale Integration, different design levels like architectural, layout, circuit level and technology optimization level are addressed [1]. In the circuit design, the proper choices of levels are used to implement combinational logic circuits for power savings. The chosen logic style influences the parameters that govern the power dissipation, switching capacitance, transition activity and short circuit currents. This paper present the multipliers designed using static logic style, dynamic logic style and constant delay logic style. The power dissipation characteristics of various multipliers using static logic, dynamic logic and constant delay logic style are compared qualitatively and with quantitatively actual logic gate implementations.

2. LOGIC STYLE

The logic style used in logic gates influences the speed, size, power dissipation, and the wiring complexity of a circuit. The circuit delay can be determined from the number of inversion levels, number of transistors in series, transistor sizes i.e., channel widths, and intra- and inter-cell wiring capacitances. The circuit *size* is dependent on the number of transistors, its size and the wiring complexity. The switching activity and the node capacitances made up of gate, diffusion, and wire capacitances are used to determine the *power dissipation which* helps to control the circuit size. The *wiring complexity* is estimated by the number of connections, their lengths and type of rail logic used [2]. These characteristics vary from one logic style to another making the proper choice of logic style crucial for circuit performance.

2.1 Requirements of Low Power

The dynamic power is expressed as

$$\mathbf{P}_{\text{dynamic}} = \mathbf{C} \, \mathbf{V}^2 \, \mathbf{f} \tag{1}$$

E-ISSN: 1817-3195

It is directly proportional to the capacitance C, supply voltage and switching frequency. The power dissipation is reduced by proper selection of these parameters. When the frequency is decreased the dynamic power also decreases with an increase in delay as frequency is inversely proportional to delay.

2.2 Static Logic Style vs Dynamic Logic Style

Static Logic styles consist of pull up and pull down network. In dynamic circuit, clock is applied to make it function in two phases, a precharge and evaluation phase [3]. In dynamic logic

<u>10th August 2014. Vol. 66 No.1</u>

© 2005 - 2014 JATIT & LLS. All rights reserved

ISSN: 1992-8645 www.jatit.org

E-ISSN: 1817-3195

style, the logic function is implemented using only n-type devices while the clock is applied to both ptype and n-type devices. In pre-charge phase, the output node is pre-charged to high, while the path to ground is turned off. In the evaluation phase, the path to high is turned off when the clock is high and path to ground is turned on. Therefore, the output node will either be high or low depending on the input [4].

An advantage of dynamic logic style is that it reduces the silicon area. Therefore static logic style requires 2n transistors and dynamic logic style requires n+2 transistors. The disadvantage of dynamic logic style is the impossibility of cascading the blocks to implement complex logic and also excessive load on the clock signal that need to be connected to every dynamic gate [5].

2.3 Constant Delay Logic Style

When CLK is high, CD logic enters predischarge period and when CLK is low, CD logic enters the evaluation period. Three modes will take place like the contention, C-Q delay, and D-Q delay modes. When CLK is low, the circuit enters into the contention mode while input remains at logic "1." Here, it experiences a temporary glitch when X is at a nonzero voltage level. When input make a transition from high to low, C-Q delay modes take place before CLK becomes low. While CLK is at low, X rises to logic "1" and Y remains at logic "0" for the entire evaluation cycle. D–Q delay mode operates at the pre-evaluated characteristic of CD logic to enable high-performance operations. In this mode, CLK falls from high to low before input transit hence X initially raises to a nonzero voltage level [6].

3. DESIGN AND ANALYSIS

The multiplier architectures are studied for high speed signal processing applications, specifications for the multiplier design, modeling the architecture, functional verification, and developing the test bench to verify the design for all possible input combinations using logic style discussed in chapter 2.

3.1 Wallace Tree Multiplier

In 1964 C. S. Wallace introduced the multiplication based on summing the partial product bits in parallel using a tree of carry save adders known as the Wallace tree [7]. This method uses a three step process to multiply two numbers.

Step 1: Assigning two 8 bit numbers as an input. Step 2: Finding the product of the bit formation. Step 3: Using compressor technique, the bit product matrix is compressed to a two row matrix by using carry save adders known as Wallace tree. Step 4: The last 12 bits are added using a ripple carry adder to produce the product.

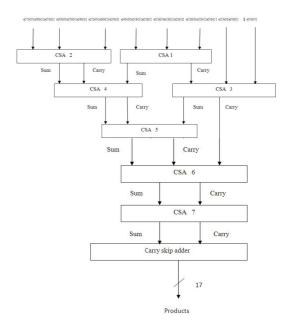


Figure 1: Block Diagram Of 8 Bit Wallace Tree Multiplier Using Static Logic Style

This method yields multipliers with delay proportional to $\log O(n)$. The block diagram of 8 bit Wallace tree multiplier is shown in Figure 1. The principle is to achieve partial product by reducing the number of bits in each column using full adders or half adders. From each column of the partial product matrix, three bits are added to produce two output bits, the sum bit in the same column and the carry bit in the next column. Hence the output matrix has been obtained by using the full adder known as the 3:2 compressors. The Wallace tree consists of numerous levels of column compressor structures till it remains with only two full width operands [9]. These two operands can then be added using regular 2N bit adders to obtain the product or 2:2 compressors respectively. Finally the partial product matrix has a depth of only two. The Wallace tree multiplier uses maximum hardware to compress the partial product matrix to obtain the final product.

The 8 bit Wallace tree multiplier using dynamic logic style is shown in Figure 2. The dynamic logic is identified when the clock goes high for which the input signal is applied to the D flip flop and output to the carry save adder. Here

<u>10th August 2014. Vol. 66 No.1</u>

© 2005 - 2014 JATIT & LLS. All rights reserved





<u>www.jatit.org</u>



three bits are added to form a partial product and produces two output bits with the sum and the carry. The sum and carry bits are applied to the next stage. The compressed bits is passed to carry propagate adder to generate product.

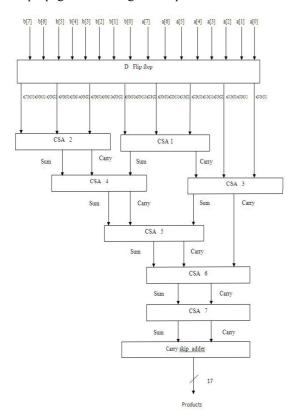
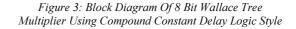


Figure 2: Block Diagram Of 8 Bit Wallace Tree Multiplier Using Dynamic Logic Style

The 8 bit Wallace tree multiplier using compound constant delay logic style is shown in Figure 3.The timing block consists of D flip flop, XOR gate and AND gate. The inputs to the XOR is the input and output of the D flip flop [7]. The clock signal and the output of XOR gate are applied to the AND gate to generate the clock signal. This signal acts as the clock signals for D flip flop. The output of timing block is applied to the carry save adder to form the partial product which is compressed to the sum and carry. Each stage of output is applied to the carry save adder till it propagates through the carry skip adder.



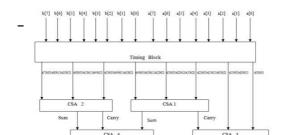
3.2 Array Multiplier

Array multiplier is based on add and shift algorithm. Each partial product is generated by the multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added.



Figure 4: Block Diagram Of 8 Bit Array Multiplier Using Static Logic Style

The 8 bit array multiplier using static logic style is shown in Figure 4. The addition can be performed with normal carry propagate adder. N-1



137

<u>10th August 2014. Vol. 66 No.1</u>

© 2005 - 2014 JATIT & LLS. All rights reserved

ISSN: 1992-8645	www.jatit.org	E-ISSN: 1817-3195
100111 1992 0010	<u>in in i</u>	1001010170170

adders are required where N is the multiplier length.

The 8 bit array multiplier using dynamic logic style is shown in Figure 5. The product of bits is applied to carry save adder at each stage. The sum and carry bits are applied to the next stage. Finally the partial products propagate through carry propagate adder to generate product [10].

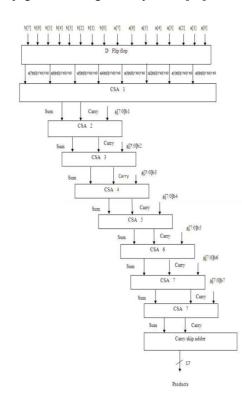


Figure 5: Block Diagram Of 8 Bit Array Multiplier Using Dynamic Logic Style

The 8 bit array multiplier using compound constant delay logic style is given in Figure 6. The data path used for the current input; the path will remain in precharge mode when the circuit switches ON. The timing block consists of D flip flop, XOR gate and AND gate. The input to the XOR is the and outputs of the D flip flop. The clock input signal and the output of XOR gate are applied to the AND gate to generate the clock signal. This signal acts as the clock signals for D flip flop. The output of timing block is applied to the carry save adder to form the partial product which is compressed to the sum and carry. The product of the bits is applied to the carry save adder. The result then propagates through the carry skip adder.

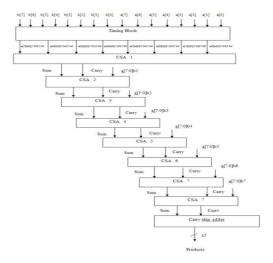


Figure 6: Block Diagram Of 8 Bit Array Multiplier Using Compound Constant Delay Logic Style

3.3 Baugh Wooley Multiplier

The Baugh wooley (BW) algorithm is a straightforward approach of performing signed multiplications. An 8-bit Baugh wooley multiplier using static logic style is shown in Figure 7, where the partial product bits have been reorganized according to Hatamian's scheme.

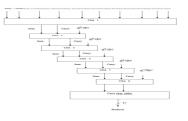


Figure 7: Block Diagram Of 8 Bit Baugh Wooley Multiplier Using Static Logic Style

The creation of the reorganized partialproduct array of an *N*-bit wide multiplier comprises three steps: *i*) The most significant bit (MSB) of the first N - 1 partial-product rows and all bits of the last partial-product row, except its MSB, are

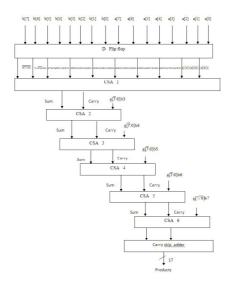
<u>10th August 2014. Vol. 66 No.1</u>

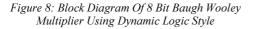
© 2005 - 2014 JATIT & LLS. All rights reserved



inverted. *ii*) A '1' is added to the N^{th} column. *iii*) The MSB of the final result is inverted.

The 8 bit Baugh wooley multiplier using dynamic logic style is shown in Figure 8. All the input signals are applied to the D flip flop and it is activated when the clock is high [10]. Furthermore, the output is applied to the carry save adder. The sum and carry bits are generated when the three bits are added .The product of bits are applied to carry save adder at each stage. The sum and carry bits are applied to the next stage. Finally the partial products propagate through carry propagate adder to generate products





The 8 bit Baugh wooley multiplier using compound constant delay logic style is shown in Figure 9. The data path used for the current input, remains in the evaluate mode. The timing block consists of D flip flop, XOR gate and AND gate. The input to the XOR is the input and outputs of the D flip flop. The clock signal and the output of XOR gate are applied to the AND gate to generate the clock signal. This signal acts as the clock signals for D flip flop. The output of timing block is applied to the carry save adder to form the partial product which is compressed to the sum and carry. The product of the bits is applied to the carry save adder and gets propagating through the carry skip adder.

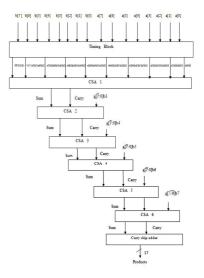


Figure 9: Block Diagram Of 8 Bit Baugh Wooley Multiplier Using Compound Constant Delay Logic Style

4. SIMULATION RESULTS

The integrated software environment (ISE) is Xilinx design software suit that allows taking the design from design entry through Xilinx device programming. The ISE project navigator manages and processes the design through varies steps in the design flow. Xilinx 9.1i provides design entry and synthesis supporting Very High speed integrated Hardware description Language (VHDL)/Verilog, place and route, completed verification and debug. The ISE design suit is the central Electronic Design Automation (EDA) product family sold by Xilinx.ISE controls all aspects of the design flow. Through the project navigator interface, we can access all the various design entry and design implementation tools.

In this paper, delay, power, power delay product and energy delay product are analyzed using Cadence tool. The analyzed data are computed for Wallace tree multiplier, Array multiplier and Baugh wooley multiplier using static logic style, dynamic logic style and constant delay logic style.

<u>10th August 2014. Vol. 66 No.1</u> © 2005 - 2014 JATIT & LLS. All rights reserved

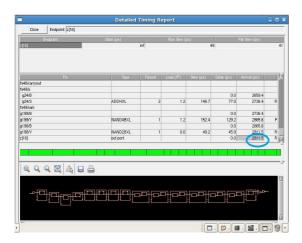
ISSN: 1992-8645

www.jatit.org

E-ISSN: 1817-3195

13:11

4.1 Results of multiplier using static logic style



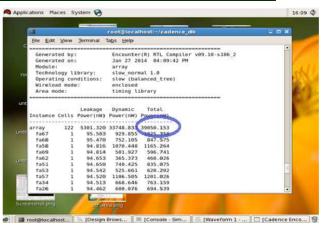


Figure 13: Power Of Array Multiplier

0AI21XI

Figure 14: Delay Of Baugh Wooley Multiplier

OAI2BB1XL

1.0 882

1.8 79.4

48.8 0.0 67.7

C... 📓 ro... 🔲 Di... 🗑 •

Applications Places System 😔

Q Q Q A A B B

Close Endpoint: 2(16)

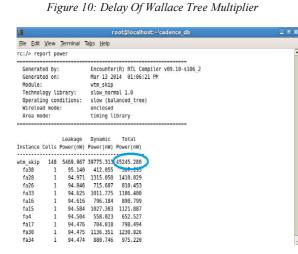


Figure 11: Power Of Wallace Tree Multiplier

	Detaile	d Timing P	teport				
Close Endpoint [2[15]							
Endpoint	Hack (ps)		Itale Since (Fall Sites (ph)	
15)		inf			66		6
fin	Tape	Fanoat	Load (ff)	Siev (pa)	Delay (p3.)	Hrick (pt)),
i68karry i69k		- Q					
654/CI		-			0.0	2472.3	_
a54/C0	ADDFX1	1	25	79.2	178.5	2650.8	E.
69karry	1.000		2.75				
70k							- 1
g54/C1					0.0	2650.8	
g54/C0	ADDFX1	1	0.0	62.0	153.6	2804.4	F
i70/carry [15]	out port	-			0.0	2804.4	
		1					
Q Q Q Q A	8						
							7
	Ppp						
				111			
				CHIC			

Applicati	ions Pl	aces Syste	m \varTheta		13:11 🔇
			100	t@localhost:~/cadence_db	
<u>Eile E</u> dit <u>V</u>	liew]€	erminal Ta <u>b</u>	s <u>H</u> elp		
Generated				t) RTL Compiler v09.10-s106_2	
Generated	d on:			01:11:28 PM	
Module:			augh_skip		
Technolog			low_normal		
Operating			low (balar	iced_tree)	
Wireload			nclosed		
Area mode	e:	3	iming libr	ary	
				Total	
	6-11-	Leakage	Dynamic Power(nW)		
instance	cetts	Power (IIW)	Power (IW)	Power (IW)	
augh skip	144	5597.679	35612.946	41210.625	
fa50	1	95.414	281.772	937 100	
fa51	1	94.824	729.398	824.223	
fa39	1	94.814	776.578	871.392	
fa41	ĩ	94.748	610.989		
			610.989 972.879		
fa41	1	94.748		705.738	
fa41 fa45	1	94.748 94.711	972.879	705.738 1067.590	
fa41 fa45 fa52	1	94.748 94.711 94.631	972.879 1176.760	705.738 1067.590 1271.390	
fa41 fa45 fa52 fa24	1 1 1	94.748 94.711 94.631 94.513	972.879 1176.760 668.646	705.738 1067.590 1271.390 763.159	
fa41 fa45 fa52 fa24 fa31	1 1 1 1 1 1 1 1	94.748 94.711 94.631 94.513 94.505	972.879 1176.760 668.646 823.250	705.738 1067.590 1271.390 763.159 917.755 480.242	
fa41 fa45 fa52 fa24 fa31 fa33	1 1 1 1	94.748 94.711 94.631 94.513 94.505 94.496	972.879 1176.760 668.646 823.250 385.747	705.738 1067.590 1271.390 763.159 917.755 480.242	

Figure 15: Power Of Baugh Wooley Multiplier

Figure 12: Delay Of Array Multiplier

<u>10th August 2014. Vol. 66 No.1</u> © 2005 - 2014 JATIT & LLS. All rights reserved TITAL

ISSN: 1992-8645

www.jatit.org

E-ISSN: 1817-3195

4.2 Results of Multiplier Using Dynamic Logic Style

	Detailed	Timing Re	port				
Close Endpoint: [z[16]							
Endpoint							
:[16]		inf		4	9)		4
fa48/carryout							_
na48.0a					0.0	2980.4	_
g24/B g24/S	ADDHXL	2	1.2	146.7	77.0	2980.4	В
924/5 a49/sum	ADDHXL	2	1.2	146.7	77.0	3057.4	- 8
a199/B					0.0	3057.4	
199/7	NAND4DNL	1	1.2	152.4	129.2	3100.0	r
196/B					0.0	3186.6	
198/V	NAND2BXL	1	0.0	49.2	45.9	3222.5	B
[16]	out port				0.0	3232.5	B
							-
Q Q Q 🕅 🙈 🖬 é	1						
a a a cut and and	=						
<u>P</u> PPPE							

Figure 16: Delay Of Wallace Tree Multiplier

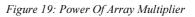
A . The second		es System	0		13:26 🔇
			root(<pre>@localhost:~/cadence_db</pre>	>
Elle Edit Vie	ew Ten	minal Ta <u>b</u> s	Help		
Wireload m	ode:	en	losed		
Area mode:		tin	ing libra	ry	
		Leakage	Dynamic	Total	
Instance	Cells	Power(nW)	Power(nW)	Power(nW)	
wtm dyn skip	165	5980.014	47107.249	53087.262	
fa31	1	95,484	1089.553	105 037	
fa38	1	95.369	547.686	643.056	
fa13	1	95.127	604.061		
fa30	1	94.795	1008.629	1103.424	
fa15	1	94.791	1132.463	1227.254	
fal4	1	94.649	709.827	804.475	
fa27	1	94.607	933.952	1028.559	
fa33	1	94.456	1148.052	1242.508	
fal6	1	94.438	970.469	1064.907	
fa26	1	94.410	664.487	758.897	
fa19	1	94.379	558.592	652.972	
fa34	1	94.377	1012.121	1106.498	
fal7	1	94.290	1104.870	1199.160	
fa28	1	94.286	1311.867	1406.152	
fa24	1	94.236	527.076	621.312	
fal2	1	94.215	503.874	598.089	
fa18	1	94.157	782.608	876.765	
fa32	1	94.142	916.145	1010.287	
100					B

Figure 17: Power Of Wallace Tree Multiplier

mislikary mislikary 0 2972.9 g64/C1 A00/7x1 1 0.0 62.0 155.6 31.6.5 F g198 soft port 0.0 2372.9 0 31.6.5 F	Endpoint Date (ps) Prior Silve (ps) Prior Silve (ps) Prior Silve (ps) (15) Inf Inf 66 Million Inf 00 2000 Million Inf 00 2754.4 Million 25 79.2 176.5 Million 25 79.2 176.5 Million 0 2972.9 F Million 0 0.0 2972.9 Million 400Fx1 1 0.0 62.0 Million 400Fx1 1 0.0 2972.9			Detaile	d Timing F	Report				
Pin Type Panet Const. (17) Ster (ps) Antrod (ps) Molitiku reference Pin Type Pinot Const. (17) Ster (ps) Antrod (ps) Molitiku reference Pinot Const. (17) Ster (ps) Antrod (ps) Molitiku reference Pinot Const. (17) Ster (ps) Antrod (ps) Molitiku reference ADD/PX1 1 2.5 79.2 17.55 297.2.6 Molitiku reference ADD/PX1 1 0.0 2.2794.4 1 0.0 2.2794.4 Molitiku reference ADD/PX1 1 0.0 6.2.0 153.6 13.65 F Molitiku reference Molitiku 0.0 2.2794.4 1 0.0 2.2794.4 1 Molitiku reference ADD/PX1 1 0.0 6.2.0 153.6 13.65.5 F Molitiku Molitiku 0.0 2.378.5 F 1 0.0 2.378.5 F	IS2 Pr/ Tope Fevore Lows (P) Tope (po) Arring (pc) <	Close	Endpoint: [2[15]							
Per Type Facod. Load (III) Sile (pz) Annol (pz) Addatury ps/d 0 2794.4 0.0 2794.4 g64/C0 ADOPNI 1 2.5 792.2 1775.5 2922.9 p66/servy ADOPNI 1 2.5 792.2 150.5 277.0 p64/C0 ADOPNI 1 0.0 620.0 150.6 277.0 p64/C10 ADOPNI 1 0.0 62.0 150.6 277.5 p64/C10 ADOPNI 1 0.0 62.0 150.6 277.5 5 p193 put pert 0.0 0.0 278.5 7	Pro Tope Fanoat Loss (17) Serve(a)) Annol (a) Million g64C0 ADDPx1 1 25 79.2 178.5 279.4 Million g64C1 ADDPx1 1 2.5 79.2 178.5 279.2 Million g64C1 ADDPx1 1 0.0 62.0 15.6 3176.5 F G64C1 ADDPx1 1 0.0 62.0 1376.5 F (15) 94.0 0.0 62.0 1365.5 F		Enapoint	Slack (ps)		Rise Stevi (Pal (lev (ps)	
MBXBary 0.0 2794.4 g\$AUC1 0.0 2794.4 g\$AUC2 0.0 2794.4 g\$AUC3 2.5 79.2 hBXBary 0.0 2794.4 hBXBary 0.0 2794.4 g\$AUC3 0.0 2794.7 g\$AUC4 0.0 279.5 g\$AUC5 0.0 279.9 g\$AUC9 ACCP21 1 0.0 g\$AUC9 ACCP21 1 0.0 279.5 g\$AUC9 ACCP21 1 0.0 207.5 10.5 g\$AUC9 ACCP21 1 0.0 207.5 10.5 10.5 g\$AUC9 ACCP21 0.0 207.5 10.5 10.5 10.5	MBCarry 00 2794.4 G\$4C1 0.0 2794.4 G\$4C2 0.0 2794.4 G\$4C3 2.5 73.2 G\$4C4 2.5 73.2 G\$4C7 0.0 2794.4 G\$4C7 0.0 279.9 G\$4C7 0.0 279.9 G\$4C7 0.0 279.9 G\$4C7 0.0 279.9 G\$4C7 0.0 210.5 G\$4C7 0.0 210.5	0210			ent.			26		6
ABORNY 00 2794.4 g&UC ADDPXI 1 2.5 79.2 79	MBCarry 00 2794.4 G\$4C1 0.0 2794.4 G\$4C2 0.0 2794.4 G\$4C3 2.5 73.2 G\$4C4 2.5 73.2 G\$4C7 0.0 2794.4 G\$4C7 0.0 279.9 G\$4C7 0.0 279.9 G\$4C7 0.0 279.9 G\$4C7 0.0 279.9 G\$4C7 0.0 210.5 G\$4C7 0.0 210.5									
ыбае убиСт убиСто мобисти убиСто ллок мобисти ллок мобисти убиСто ллок мобисти убиСто ллок мобисти убиСто ллок мобисти убиСто ллок мобисти убиСто мобисти убиСто ллок мобисти убиСто мобисти убиСто мобисти убиСто мобисти убиСто мобисти убиСто мобисти убиСто мобисти убиСто	Mile 00 27944 6400 ADDFx1 1 2.5 79.2 178.5 2872.9 F Mile 00 2974.4 1 2.5 79.2 178.5 2872.9 F JOIk 00 2972.4 00 2972.9 F 00 2972.9 F JOIk 00 62.0 155.6 318.5 F OVerary 119 0.0 120.5 F JIS 9xl port 0.0 136.5 F IS 136.5 F				Fariout	Load (IF)		Delay (po.)		
op/sect 0.0 2794.4 P dbltamy ADDFXI 1 2.5 79.2 178.5 292.9 P dbltamy ADDFXI 1 2.5 79.2 178.5 292.9 P dbltamy ADDFXI 1 2.5 79.2 178.5 297.9 P dp4Dr1 ADDFXI 1 0.0 62.0 150.6 37.85 F Vibamy Vibamy 0.0 62.0 150.6 37.85 F (HS) 94 port 0.0 62.0 0.0 13.85 F	g\$4C1 ADDFx1 1 25 792 1785 29724 F M80xmy ADDFx1 1 25 792 1785 29724 F M80xmy ADDFx1 1 25 792 1785 29724 F g\$4C1 ADDFx1 1 0.0 €2.0 151.6 3125 F y%2my ADDFx1 1 0.0 €2.0 151.6 3125 F 115 904 port 0 0 31685 F									
65400 ADDFX1 1 2.5 79.2 178.5 2872.9 F 70% 70% 0 0 2072.9 F 65401 0 0.0 2972.9 F 65401 0 0.0 2972.9 F 65401 0 0.0 2972.9 F 70% 70% 0.0 2072.9 F 70% 94 0.0 2072.9 F 70% 94 0.0 208.9 F 70% 94 0.0 20.	65400 ADDPRI 1 2.5 79.2 178.5 2972.9 F 70% 5601 0 2972.9 0 5602 0.0 2972.9 F 65400 ADDPRI 1 0.0 62.0 153.6 3178.5 F 70% SH port 0.0 10.5 7 1 0.0 10.5 F 70% SH port 0.0 10.6,5 F 1 <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td>									
Jebevry Job 0 2972.9 JefCit JobPrit 1 0.0 6207.9 Jebevry ADDPrit 1 0.0 6207.9 JefCit Jia 5 1 1 1 JefCit Jia 5 1 1 1 JefCit Jia 5 1 1 1 0.0 1 JefCit Jia 5 1 1 0.0 1 1 1 0.0 1	Jebevny Jebevny <t< td=""><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></t<>									
70% SeCi SeCi SeCi SeCi SeCi SeCi SeCi SeCi	706 FACI FACO 706 FACO 707 707 708 707 707 707 707 707			ADDFX1	1	2.5	79.2	178.5	2972.9	F
μρείο' μοργί 0 2072 β γείο' μοργί 0 62.0 153.6 1328 f γοίανην 0.0 62.0 153.6 1328 f 1 γοίανην 0.0 0.0 133.6 1 1 1 1 0.0 1	(54C) 00 2272 9 54CO ADDPXI 1 00 620 1555 13155 F 700ary 94 port 0.0 (31255 F 15] 94 port 0.0 (31255 F				-					_
APECO ΑΟΓΡΑΙ 1 00 62.0 153.6 33265 F TORMY 844 port 0.0 33265 F 153 844 port 0.0 33265 F 154 0.0 33265 F 154 0.0 33265 F 154 0.0 33265 F 154 0.0 154 0.	Observe ACOPNI 1 0.0 62.0 155.6 3128.5 F 153 sod port 0.0								00000.0	-
700any 16] put port 0.0 (3125) r	700aary 16] 94 port 0.0 1265 r			ADDEVI			62.0			
19]	ing outport 0.0 12635 r			ADDIAL		0.0	62.0	133.0	3120.5	-
				out port				0.0	1126.5	S F
		a Q	9. 63. 44. 67.	3)		1			

Figure 18: Delay Of Array Multiplier

Applica	tions I	Places Syst	tem 😔		31:41 🤇
			1/0	ot@localhosts=/cadence_db	
Eile Edit	View .	Jerminal Ta	abs Help		
Generate Generate Module: Technold Operatir Wireload Area mod	nd by: nd on: ogy lib ng cond d mode: de:	orary: Hitlons:	Encounter Feb 12 20 array_dyn slow_norm	al 1.0 anced_tree)	
		Leakage	Dynamic	Total	
Instance	Cells	Power(nW)	Power(nW)	Power(nW)	
array dyn	138	5070 733	41437.831	47316 564	
fa67	1.50	95.614	1096.182	1401 702	
Tabl		95.117	960.623	1055.740	
1358		95.036	879.502	974.537	
fa57	î	94,772	958,409	1053.181	
fa33	1	94,693	678.510	773.203	
1035	ĩ	94.667	717.434	812.101	
1042	ĩ	94.636	793.146	887.782	
1869	1	94,495	735.727	830.222	
1049	1	94.466	1032.967	1127.433	
Tase	1	94,434	937.343	1031.777	
1019	1	94,405	434.233	528.638	
Ta26	1	94.304	579.759	674.143	
1032	1	94.382	521.208	615.590	
fal8	1	94.380	468.446	562.826	
Ta51	1	94.365	923.132	1017.498	
1044	1	94.323	843,708	938,032	



Siberront Display	Endpoint Stack (pt) Pairs (Stark (pt)) Pair (Stark (pt)) Pair (Stark (pt)) 6) Imm Stark (pt)) Pairs (Stark (pt)) Pairs (Stark (pt)) 4490 Imm Stark (pt)) Stark (pt)) Destroy (pt)) Pairs (pt)) 4490 Imm Stark (pt)) Stark (pt)) Destroy (pt)) Destroy (pt)) Stark (pt)) 4490 OA2BB134 2 Imm Stark (pt)) Destroy (pt)) Stark (pt)) 56xerytini 0 OA2BB134 2 Imm 0 31622 F 56xerytini 0 0 Stark (pt)) 0 31622 F 59xerytini 0 0 0 31622 F 31622 F 59xerytini 0 0 0 31622 F 3162 F 59xerytini 0 0 0 3162 F 32xe3 F 59xerytini 0 0 0 0 3162 F 59		Detailed	Timing Re	sport				
Nn Type Ferrout Loxet (fT) Seer (ps) Delay (ps) Artinat (ps) 49800 0.0 31145 1 0.0 31145 1 6807 0.0.42861XL 2 1.8 79.4 67.7 3162.5 6907 0.0.42861XL 2 1.8 79.4 67.7 3162.5 308/mm 0.0.42861XL 2 2.1 66.4 147.1 3328.3 1 309/mm 0.02XL 2 2.1 66.4 147.1 3328.3 1 70A 0.0 38.5 11.7.8 346.62 1 0.0 3328.3 1	Implementation Type Percont Lood (fF) Save (µs) Delay (cs) Antro (cs) <th>Close Endpoint: [2[16]</th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th> <th></th>	Close Endpoint: [2[16]							
No. Type Percol. Lood (17) Serv (pp.) Defin, (co.) Mith (co.)	Prod Type Percol. Load (IT) See (ps) Delay (cs) Permit (cs) 480V 0.02801 XL 2 1.8 79.4 0.7 316.2 1 480V 0.02801 XL 2 1.8 79.4 0.7 316.2 1 98xmont 0.0 9116.5 0.0 316.2 1 38xmont 0.0 0.0 316.2 1 0.0 316.2 1 39xmont 0.0 0.0 316.2 1 0.0 316.2 1 70A 0.0 0.0 338.5 1.7 3440.8 1 94 port 0.0 38.5 1.75 3440.8 1 94 port 0.0 38.5 1.75 3440.8 1								
Me800 0 0.0 311.4 S BAY 0.028B1XL 2 1.8 79.4 67.7 3162.2 Sidkarryont 2 1.8 79.4 67.7 3162.2 1 Sidkarryont 2 2.1 66.4 1.0 3162.2 3162.2 Sidkarryon 0 2 2.1 66.4 1.01 3162.2 3162.2 Sidkarryon 0 2 2.1 66.4 1.01 3162.2 1.0 Sidkarryon 0 2 2.1 66.4 1.01 3162.2 1.0 Sidkarryon 0 2 2.1 66.4 1.01 326.3 1.0 70A 0 0 0.00 38.5 1.16 3.0 <	4960 0.0 91145 0.0 91145 1 95% 0.7 9182 1.8 794 67.7 9182 1 85% 0.0 91145 1 794 67.7 9182 1 85% 0.0 9182 1 0.0 9182 1 0.0 9182 1 35% 0.0 0.0 3283 F 1 0.0 3283 F 70A 0.0 0.0 385 1175 34463 F 6 put port 0.0 385 1175 34463 F	16)		inf		2	2		
Me800 0 0.0 311.4 S BAY 0.028B1XL 2 1.8 79.4 67.7 3162.2 Sidkarryont 2 1.8 79.4 67.7 3162.2 1 Sidkarryont 2 2.1 66.4 1.0 3162.2 3162.2 Sidkarryon 0 2 2.1 66.4 1.01 3162.2 3162.2 Sidkarryon 0 2 2.1 66.4 1.01 3162.2 1.0 Sidkarryon 0 2 2.1 66.4 1.01 3162.2 1.0 Sidkarryon 0 2 2.1 66.4 1.01 326.3 1.0 70A 0 0 0.00 38.5 1.16 3.0 <	4960 0.0 91145 0.0 91145 1 95% 0.7 9182 1.8 794 67.7 9182 1 85% 0.0 91145 1 794 67.7 9182 1 85% 0.0 9182 1 0.0 9182 1 0.0 9182 1 35% 0.0 0.0 3283 F 1 0.0 3283 F 70A 0.0 0.0 385 1175 34463 F 6 put port 0.0 385 1175 34463 F								
HBW DA2BB1XL 2 1.8 79.4 67.7 3182.2 I SBAxmont 2 1.8 79.4 67.7 3182.2 I SBAxmont 2 1.8 79.4 67.7 3182.2 I SBAxmont 2 2.1 66.4 147.1 3328.3 I SBAxmont 2 2.1 66.4 147.1 3328.3 I SBAxmont 1 0 88.5 11.75 3446.2 70A Ph2XL 1 0.0 88.5 11.75 3446.2	de/v DA26B1%L 2 1.8 79.4 67.7 3162.2 F Stearyon 390.4 2 1.8 79.4 67.7 3162.2 F Stearyon 390.4 0 2 1.8 79.4 67.7 3162.2 F Stearyon 0P2XL 2 2 66.4 147.1 3328.3 F Stearyon 0P2XL 2 2.1 66.4 147.1 3328.3 F 700v 0P2XL 1 0.0 385 117.6 3446.8 F 61 outport 1 0.0 385 117.6 3446.8 F		Туре	Fanout	Load (fF)	Siew (ps)			
Siberront Display	Biblestrout		04/2881 21	2	1.0	70.4			
Stearryn DR2XL 2 2.1 66.4 147.1 3328.3 1 Stearryn DR2XL 2 2.1 66.4 147.1 3328.3 1 Stearryn DR2XL 2 2.1 66.4 147.1 3328.3 1 T0A T0A T04.5 T04.5 T05.3 1	Signaryon 0 0 0 3162 2 Signaryon 0R2XL 2 2 66 147.1 33283 3 F Signaryon 0R2XL 2 2.1 66.4 147.1 33283 3 F TOA 00 3828 0.0 33283 4 F		UALZBBIAL	2	1.0	79.4	67.7	3102.2	
380A 0 00 3182.2 380Y 0R2XL 2 2.1 664 147.1 3328.3 1 380xyrout 0 328.4 0.0 3328.3 1 70A 0 0.0 3328.3 1 0.0 3328.3 1 70AY 0R2XL 1 0.0 385 117.8 3466.4 1 1 1 0.0 385 117.8 3466.4 1 1 0.0 385 117.8 3466.4 1 1 0.0 385 117.8 3466.4 1 1 0.0 385 117.8 346.6 1	39A 00 010 3102.2 1 66.4 147.1 3329.3 9K-syroud 0R2XL 2 2.1 66.4 147.1 3329.3 70A 0R2XL 1 0.0 3329.3 1 1.7 3349.7 1 70Y 0R2XL 1 0.0 3829.3 1 1.7 3448.9 F 61 0.0 3448.9 F 0.0 3448.9 F								
ORAX DR2XL 2 2.1 66.4 147.1 3328.9 1 700A 0 0 0 0 3328.9 1 700A 0 0 85.5 11.75 3466.2 1	39/V OR2XL 2 2.1 66.4 147.1 3329.3 F 70/A 00 3329.3 1 0.0 3329.3 F 70/V 0P2XL 1 0.0 3329.3 F F 6 147.1 3469.3 F 6] pat.port 1 0.0 385 117.6 3469.3 F						0.0	3182.2	
70/A 0.0 3329.3 70/V DR2XL 1 0.0 38.5 117.6 3449.9 1	700A 0.0 3329 7 707V 0R2XL 1 0.0 385 175 3446.8 7 61 001 port port 0.0 385 175 3446.8 7		OR2XL	2	2.1	66.4	147.1		F
70/V DR2XL 1 0.0 38.5 117.6 3446.9 F	OR2XL 1 0.0 38.5 117.6 3446.9 r 61 0.0 34.5 0.0 3446.9 r	59/carryout							
		70/A					0.0	3329.3	
		70/V		1	0.0	38.5			F
		16]	out port				0.0	3446.9	F
			-비난민빈				1 1 1111		
9°EEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEEE									

Figure 20: Delay Of Baugh Wooley Multiplier

Applications			-		14:32 🔇
			root@lo	calhost:~/cadence_db	
Eile Edit View	/ Termin	nal Ta <u>b</u> s <u>H</u>	elp		
Generated by	v:	Encou	inter(R) RT	L Compiler v09.10-s106 2	
Generated of	n:	Mar 1	18 2014 02	:31:42 PM	
Module:		baugh	dyn skip		
Technology	library	: slow	normal 1.0		
Operating c	ondition	ns: slow	(balanced	tree)	
Wireload mon	de:	encle	osed		
Area mode:		timir	ng library		
		Leakage	Dynamic	Total	
Instance	Cells	Power(nW)	Power(nW)	Power(nW)	
baugh_dyn_ski			41388.50		
fa50	1	95.822	225.151	320.973	
fa29	1	94.766	654.164	748.929	
fa52	1	94.745	1222.083	1316.827	
fa46	1	94.707	982.897	1077.604	
fa23	1	94.693	700.582	795.275	
fa43	1	94.540	1168.329	1262.870	
fa49	1	94.507	423.187	517.695	
fa45	1	94.504	922.934	1017.438	
fa53		94.467	1064.791	1159.258	
fa33	1	94.450	703.264	797.714	
fall	1	94.405	434.235	528.641	
fa39	1	94.399	845.679	940.078	
fa44	1	94.398	966.178	1060.576	

Figure 21: Power Of Baugh Wooley Multiplier

10th August 2014. Vol. 66 No.1

 $\ensuremath{\mathbb{C}}$ 2005 - 2014 JATIT & LLS. All rights reserved $^{\cdot}$

ISSN: 1992-8645

www.jatit.org

E-ISSN: 1817-3195

14:53

79.4

4.3 Results of Multiplier Using Compound Constant Delay Logic Style

Applications Places Syster	Detailed 1	timina Pr	mort			14:37 🔇	Applica	tions F	laces Syst	tem \Theta		11:41
Close Endpoint:[z[16]	Detailed	nining ike	pore							19	ot@localhost:~/cadence_db	
			100000000000000000000000000000000000000				Ele Edit	View 1	erminal Ta	abs <u>H</u> elp		
Endpoint		inf	Post Dev (pr	49		ul liner (ps) 41	Generate Generate Module: Technolo Operatin Wireload	d on: gy lit g cond		Feb 12 20 array_dyn slow_norm	(R) RTL Compiler v09.10-s106_2 14 11:38:24 AM al 1.0 anced_tree)	
Pin.	Type	Percet	tant(III)	Sev (p3) 6	Millio (\$113)	Arrest (co)	Area mod	le:		timing li	brary	
48/carryout 49/a												
459. (24/8					0.0	2991.6						
j24/5	ADDHXL	2	12	146.7	77.0	3068.6 R	114010-0000-0000		Leakage	Dynamic	Total	
49/sum	Par Par Inda	-		140.1	1.1.30		Instance	Cells	Power(nW)	Power(nW)	Power(nW)	
55/0					0.0	3068.6					A DECEMBER OF THE OWNER OF	
55/Y	NAND4BXL	1	1.2	152.4	129.2	3197.8 F	array_dyn	138			47316.564	
54/8					0.0	3197.8	fa67 fa68	1	95.614 95.117	1096.182 960.623		
54/V	NAND28XL	1	0.0	49.2	45.9	3242.2 R	fa58	1	95.036	879.502	974,537	
6]	out port		100		0.0	3243.7 R	fa57	1	94.772	958.409		
							fa33		94,693	678.510	773.203	
							fa35	- 1	94.667	717.434	812.101	
and the state	AN CONTRACTOR		100-010			a tool tool togg	fa42	î	94.636	793.146		
	m						1a69	ĩ	94,495	735,727	830,222	
							fa49	1	94,466	1032.967		
						72	Ta50	1	94,434	937.343	1031.777	
							fa19	1	94.405	434.233	528.638	
	plaiala					a li albo	fa26	1	94.384	579.759	674.143	
			- HOL	linelic	PUTCH		fa32	1	94.382	521.208	615.590	
		tt		14			fa18	1	94.380	468.446	562.826	
							fa51	1	94.365	923.132		
						1.1	fa44	1	94.323	843.708	938.032	
						12	🔿 🗐 Iro	at (b) o c	albort - /ca	CT IC M	ence Encounter(🗃 root@localhost.~/ca	Codence Encounter
							a fu	nue nuc	an ioac m/ca.	in Juni Juni	ence Encountert	The former of Eliconities (11

Figure 22: Delay Of Wallace Tree Multiplier

Figure 25: Power Of Array Multiplier

ces System 💡

9. 22 4 8 8

Applications I	laces	System 😪)				14:36 🔇
a			rootalo	alhost:~/c	adence_db		2 n X
Ele Edit View	fermina	l Tabs He	lp.				
Synthesis succ	eeded.						E
rc:/> report pow	/er						
Generated by:		Encour	ter(R) RT	Compiler	v09.10-s10	6.2	
Generated on:			2014 02				
Module:			ce skip cd				
Technology lib	rary:		ormal 1.0				
Operating cond		stow (balanced	tree)			
Wireload mode:		enclos	sed				
Area mode:		timing	1 Library				
***************	*****				*******		
		1000000	1211111	2000			
Instance	Colle	Leakage Power(nW)	Dynamic Rowor(ow)	Total			
Tustance	cetts	rower (nn)	rower(im)	FOWER THW)	222		
wallace skip cd	214	6480.236	6632 476	13112.705			
fal	1	91.015	0.000				
fa10	- 2	91.015	0.000	91.015			
fall	ĩ	91.015	0.000	91.015			
fa12	- î	91.015	6.000	91.015			
fa13	1	91.015	0.000	91.015			
fa14	1	91.015	0.000	91.015			
fa15	1	91.015	0.000	91.015			
fal6	1	91.015	0.000	91.015			
fal7	1	91.015	0.000	91.015			1
Ta18	1	91.015	0.000	91.015			1
fa19	1	91.015	0.000	91.015			
fa2	1	91.015	0.000	91.015			
fa20	1	91.015	0.000	91.015			
, fa21	1	91.015	0.000	91.015			
							(

Figure 23: Power Of Wallace Tree Multiplier

		Detailer	a Timing A	leport				
Close	Endpoint: dff31/q_reg	Ø						
	Endpoint							
1731 <i>/a_rrra</i> D			inf			80		6
	Etti (Jag #	Farott	Losd (IT)	(liew (p.s.)		Arrivis (p3)	/
g54/C1		10000	71	1.000		0.0	2775.3	
g54/C0 x69/carry		ADDFX1	1	2.5	79.2	178.5	2953.8	F
etsicarry 70/c								_
154/CI			-			0.0	2953.8	
54/C0		ADDFX1	1	1.1	69.3	165.3	3119.1	F
70/carry		Paratici			00.5	100.0	01101	- 1
fr31./d								- 1
L reg/D		DFFQXL				0.0	3110.1	
a_reg/CK		retup			0.0	85.8	3204.5	A I
। ब् Q	۱ ۹ ۲ ۸ ۵	8						
			lof					

Figure 24: Delay Of Array Multiplier



Application:	Applications Places System 🚱			14:53	
4			roote	localhost:-/cadence_db	
Ble Edit View	w Jerm	inal Tabs	Help		
Generated b	ev:	Enco	unter(R)	RTL Compiler v09.10-s106 2	
Generated o	n:	Mar	18 2014	02:52:53 PM	
Module:		baut	h cd skip		
Technology	library	Y: \$10	v normal 1	.0	
Operating c	onditio	ons: slow	(balance	d tree)	
Wireload mo	de:	encl	losed		
Area mode:		timi	ing librar	Y	
			S. 5.	22122	
142010-00100000	Nanadir	Leakage	Dynamic	Total	
Instance	Cells	Power(nW)	Power(nW)	Power (nid)	
	209			13820,859	
augh_cd_skip faS0		6702.715	7118.14	13820.859	
1451	1	96.048	378,136	474, 184	
fa52	-	90.048	36.298	127.844	
fal0		91.015	0.000	91.015	
1011		91.015	0,000	91.015	
fa12		91.015	0.000	91.015	
fal3		91.015	0.000	91,015	
fal4		91.015	0.000	91.015	
fa15		91.015	0.000	91.015	
fa16	1	91.015	0,000	91.015	
1017	1	91.015	0.000	91,015	
1a18	1	91.015	0.000	91.015	
fa19	1	91.015	0,000	91.015	
fa20	1	91,015	0,000	91.615	
1021	1	91.015	0.000	91.015	
1a22	1	91.015	0.000	91.015	
1.423	1	91.015	0,000	91.015	



10th August 2014. Vol. 66 No.1

© 2005 - 2014 JATIT & LLS. All rights reserved

ISSN: 1992-8645

www.jatit.org

4.5 Comparison table

4.4 Performance analysis

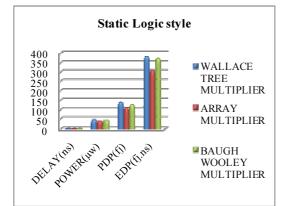


Figure 28: Performance Analysis Of Multipliers Using Static Logic Style

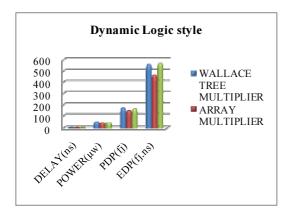


Figure 29: Performance Analysis Of Multipliers Using Dynamic Logic Style

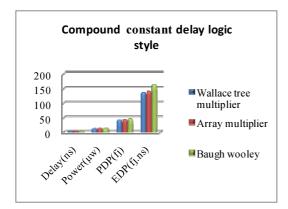


Figure 30: Performance Analysis Of Multipliers Using Compound Constant Delay Logic Style

Types of multiplier	Delay (ns)	Power (μw)	PDP (fj)	EDP (fj.ns)
Wallace tree multiplier	2.911	45.245	131.1	381.63
Array multiplier	2.804	39.050	109.4	306.75
Baugh wooley multiplier	3.006	41.210	123.87	372.35

Table	2.	Dyna	mic	Logic	Style.
ruoic	4.	Dyna	mic	LUSIC	Siyic

Types of multiplier	Delay (ns)	Power (µw)	PDP (fj)	EDP (fj.ns)
Wallace tree multiplier	3.232	53.087	171.57	554.51
Array multiplier	3.126	47.316	147.90	462.33
Baugh wooley multiplier	3.446	47.567	163.91	564.83

Table 3: Compound Constant Delay Logic Style

Types of multiplier	Delay (ns)	Power (µw)	PDP (fj)	EDP (fj.ns)
Wallace tree multiplier	3.243	13.112	42.5	137.89
Array multiplier	3.204	13.868	44.43	142.35
Baugh wooley multiplier	3.452	13.823	47.70	164.66

5. CONCLUSION

Multipliers are the major portions in hardware consumption for filters. Carry skip adder is used to speed up the accumulation. Wallace tree multiplier reduces the delay by taking partial product reduction method. Wallace tree multiplier, array multiplier and Baugh wooley multiplier are implemented using static logic style, dynamic logic style and Compound constant delay. From the tabulated results it is clear that the Energy Delay product for a Wallace Tree Multiplier is 137.89 fj.ns, Array multiplier is 142.35 fj.ns and the Baugh Wooley Multiplier is 164.66 fj.ns. The simulation result shows that energy delay product of Wallace tree multiplier, array multiplier and Baugh wooley multiplier using compound constant delay logic style is better than static logic style and dynamic logic style.

E-ISSN: 1817-3195

<u>10th August 2014. Vol. 66 No.1</u>

 $\ensuremath{\mathbb{C}}$ 2005 - 2014 JATIT & LLS. All rights reserved $^{\cdot}$

ISSN: 1002 9645	www.istit.ana	E ICCN, 1017
ISSN: 1992-8645	www.jatit.org	E-ISSN: 1817-3

REFERENCES:

- M. Aguirre-Hernandez and M. Linares-Aranda, "CMOS full-adders for energyefficient arithmetic applications," *IEEE Trans. Very Large ScaleIntegr. (VLSI) Syst.*, vol. 19, no. 99, Apr. 2010 pp. 1–5.
- [2] Douglas A.Pucknell and Eshraghian, Basic VLSI Design, 3rd ed. Reading, PHI, 2010
- [3] N.Goncalves and H.De Man, "NORA: A racefree dynamic cMOS technique for pipelined logic structures," *IEEE J.Solid –State Circuits*,vol.18, no.3, Jun.1983, pp261-266.
- [4] A.K.Kumar,D.Somasundareswari,T.S.Pradeepa ,"Design of Low Power Multiplier with Energy Efficient Full adder using DPTAAL,"*Hindawi Publishing Corporation Very Large Scale Integr.(VLSI) Syst.*, vol.2013, Feburary 2013 Article ID 157872, pages 9.
- [5] C.Lee and E.Szeto, "Zipper CMOS," *IEEE Circuits Syst.Mag.*, vol .2, no. 3, May 1986,pp 10 -16.
- [6] V.Navarro-Botello, J.A.Monitel-Nelson, and S.Nooshabadi,"Low Power Arthimetic Circuit in Feedthroughdynamic CMOS logic," in *Proc.IEEE Int.* 49th Midw. Symp.Circuits Syst., Aug.2006, pp. 709-712.
- [7] N.Wesye and D.Harris, CMOS VLSI Design: A Circuits and Systems Perspective, 4th ed. Reading, MA;Addison Wesley, Mar. 2010.
- [8] L.Raja,B.M.Prabhu and K.Thanushkodi 'Design of Low power digital multiplier using Dual Threshold voltage Adder Module, "International Conference on Communication and System Design 2011".
- [9] M.Ravindrakumar, G.ParameswaraRao, "Design and Implementation of 32 bit High Level Wallace Tree Multiplier," *International Journal of Technical Research and Applications*, vol. 1 Issue 4 (sept-oct 2013),pp.86-90.
- [10] Rafati, S. Fakhraie and K.Smith, "A 16 bit barrel shifter implemented in data driven dynamic logic(D3L)," *IEEE Trans.Circuits Syst.I,Reg.Papers*,vol.53, no.10, Oct 2006, pp. 2194-2202.
- [11] M.Sinangil, N.Verma and A. Chandrasekaran, "A Reconfigurable 8T Ultra Dynamic Voltage Scalable(U-DVS) SRAM in 65 nm cMOS",*IEEE Journal in solid- State Circuits* vol. 44 Nov 2009, pp 3163- 3173.

[12] N. Verma and A. Chnadrakasn, 'A 65 nm 8T Sub-vt SRAM Employing Sense –Amplifier Redundancy," *IEEE International in Solid State Circuits Conference, 2007. ISSCC* 2007.Digest of Technical Papers., 11 -15 2007, pp 328 606.

-3195

[13] Zimmermann and W.Fichtner, "Low power logic styles: CMOS versus pass transistor logic," *IEEE Journal of Solid State Circuits* vol 32 July 1997, pp 1079-1090.