LOGIC OPTIMIZATION USING TECHNOLOGY INDEPENDENT MUX BASED ADDERS IN FPGA

R.Uma and P.Dhavachelvan

Department of Computer Engineering, School of Engineering, Pondicherry University, Pondicherry, India
uma.ramadass1@gmail.com
dhavachelvan@gmail.com

ABSTRACT

Adders form an almost obligatory component of every contemporary integrated circuit. The prerequisite of the adder is that it is primarily fast and secondarily efficient in terms of power consumption and chip area. Therefore, careful optimization of the adder is of the greatest importance. This optimization can be attained in two levels; it can be circuit or logic optimization. In circuit optimization the size of transistors are manipulated, where as in logic optimization the Boolean equations are rearranged (or manipulated) to optimize speed, area and power consumption. This paper focuses the optimization of adder through technology independent mapping. The work presents 20 different logical construction of 1-bit adder cell in CMOS logic and its performance is analyzed in terms of transistor count, delay and power dissipation. These performance issues are analyzed through Tanner EDA with TSMC MOSIS 250nm technology. From this analysis the optimized equation is chosen to construct a full adder circuit in terms of multiplexer. This logic optimized multiplexer based adders are incorporated in selected existing adders like ripple carry adder, carry look-ahead adder, carry skip adder, carry select adder, carry increment adder and carry save adder and its performance is analyzed in terms of area (slices used) and maximum combinational path delay as a function of size. The target FPGA device chosen for the implementation of these adders was Xilinx ISE 12.1 Spartan3E XC3S500-5FG320. Each adder type was implemented with bit sizes of: 8, 16, 32, 64 bits. This variety of sizes will provide with more insight about the performance of each adder in terms of area and delay as a function of size.

KEYWORDS

Technology independent mapping, Adder topologies, FPGA, Multiplexer based Adders, Logical effort, Delay calculation

1. INTRODUCTION

In most of the digital systems, adders are the fundamental component in the design of application specific integrated circuits like RISC processors, digital signal processors (DSP), microprocessors etc. The design criterion of a full adder cell is usually multi-fold. Transistor count is, of course, a primary concern which largely affects the design complexity of many function units such as multiplier and Arithmetic logic unit (ALU). The basic principle in designing digital adder circuit hovers around reducing the required hardware thus reducing the cost too. To achieve this, logical optimization helps to obtaining minimum number of literals to minimizing the transistor count and the power consumption and increasing the speed of operation.

A logic expression can be expressed in various logic forms, which differ in literal counts. In widely-used MOS circuits, the number of transistors to implement a Boolean expression is directly proportional to literal counts in its logic form[1-2]. Thus, a logic optimization is simply to derive a logic form with the fewest literals. Logic level optimization is the design task where an RTL circuit description is optimized in terms of area, delay and power.

DOI : 10.5121/vlsic.2012.3412

Conventionally, a logic level optimization can be achieved in two steps; they are Technology Independent (TI) and Technology – Dependent (TD) optimization. In the former method the circuit's Boolean description is optimized ignoring the technology in which the circuit will be implemented. In the second method, the output of the technology independent optimization step (i.e. optimized Boolean network) is optimized considering the adopted technology. During the TI step there is much flexibility to restructure circuit logic to minimize the number of nodes and literals, thereby reducing the area of the circuit. During this stage the circuit can be most effectively restructured to meet the specified delay constraints critical for circuit performance. During the TD step, the delay characteristics of the target library are available, but very few restructuring of the circuit is possible.

Logical effort [13, 14] has been widely used in a variety of application domains as well as in industry standard EDA synthesis tools. Designing a circuit to achieve the greatest speed or to meet a delay constraint presents a bewildering array of choices [13, 14]. The method of logical effort is a design procedure for achieving the least delay along a path of a logic network. This method is based on a simple approximation that treats MOS circuits as networks of resistance and capacitance. This RC model provides simple mathematical calculation to obtain the circuit's maximum speed. In this paper the delay model for optimized full adder circuit and its delay estimation is also presented.

In this paper, we proposed 20 different Boolean expressions (logic construction) to implement a 1-bit full adder circuit. All the Boolean expressions are realized in terms of CMOS logic. The optimization method used in this work is technology independent optimization step. These Boolean logic realization and performances are analyzed in terms of transistor count, delay and power dissipation using Tanner EDA with TSMC MOSIS 250nm technology. From this analysis the optimized equation is selected and it is implemented in terms of multiplexers and it is incorporated in selected existing adder topologies like ripple carry adder, carry look-ahead adder, carry skip adder, carry select adder, carry increment adder and carry save adder and its performance is analyzed in terms of area (slices used) and maximum combinational path delay as a function of size. Performance comparison of existing and logic optimized schemes are analyzed on cell-based VLSI technologies, such as standard-cell based FPGAs. The cell-based approach is justified by its wide-spread use in the ASIC design community and its compatibility with hardware synthesis, which in turns satisfies the demand for ever higher productivity. This work presents the significance of adder comparison in terms of CLBs occupied and its maximum combinational delay exist in adder topology.

The organization of the paper is as follows: The section 2, describes the existing adder topologies. Section 3, presents the mathematical Boolean expression for the design of 1-bit full adder cell. Section 4 presents the simulation and analysis of full adder using Tanner EDA. Section 5 presents the FPGA implementation of different adder topologies. Section 6 gives the summary of comparison. Finally the conclusion is presented in section 7.

2. REVIEW OF EXISTING ADDER TOPOLOGY

Most of the VLSI applications, such as digital signal processing, image and video processing, and microprocessors, extensively use arithmetic operations. Addition, subtraction, multiplication, and multiply and accumulate (MAC) are examples of the most commonly used operations. The 1-bit full-adder cell is the building block of all these modules. Thus, enhancing its performance is critical for enhancing the overall module performance. This section presents the overview of the existing adder topologies.

In FPGAs, the most fundamental component implemented for high speed applications like microprocessors, arithmetic logic unit, program counters and multiply accumulate unit. Lot of

implementations has been made for these adder topologies for optimizing area, delay and power dissipations. In the reference [1], it provides an overview for the comparison of adders in the early design phase for selecting their appropriate design structure for implementing adders with the constraints of area, delay and power dissipation. This paper also reveals the pre-estimation of energy-delay, product, energy-delay estimation and power estimation in the energy delay space. In the reference [2], the proposed high speed and low power full adder cells which has designed with pass transistor logic styles to reduce the power delay product (PDP). This paper also reports the performance comparison of adder cells with CMOS, DCVS, CPL, DPL, Swing restorer CPL and hybrid styles. This paper shows the implementation of adder cells with enhanced carry generation stage which is implemented with multiplexes. This feature provides that for this logic there are no internal signals being generated for controlling the selection of output multiplexers, thereby reducing the full voltage swing, delay and overall propagation delays.

The adder topology is present in literature [3-12], Ripple Carry Adder (RCA) is the simplest, but slowest adders with O(n) area and O(n) delay, where n is the operand size in bits. Carry Look-Ahead (CLA) have O(nlog(n)) area and O(log(n)) delay, but typically suffer from irregular layout. On the other hand, Carry Skip Adder, carry increment and carry select have O(n) area and $O(n^{l+2/l+1})$ delay provides a good compromise in terms of area and delay, along with a simple and regular layout. Carry save adder have O(n) area and O(log n) delay. The ripple carry adder, the most basic of flavours, is at the one extreme of the spectrum with the least amount of CLBs but the highest delay. CLA adders can be realized in two gate levels provided there is no limit on fan in/out. The carry select adders reduce the computation time by pre-computing the sum for all possible carry bit values (ie '0' and '1'). After the carry becomes available the correct sum is selected using multiplexer. Carry select adders are in the class of fast adders, but they suffer from fan-out limitation since the number of multiplexers that need to be driven by the carry signal increases exponentially. In the worst case, a carry signal is used to select n/2 multiplexers in an n-bit adder. When three or more operands are to be added simultaneously using two operand adders, the time consuming carry propagation must be repeated several times. If the number of operands is 'k', then carries have to propagate (k-1) times.

3. MATHEMATICAL EQUATIONS FOR FULL ADDER

A full adder is a combinational circuit that performs the arithmetic sum of three bits: A, B and a carry in, C, from a previous addition produces the corresponding SUM, S, and a carry out, CARRY. The various equations for SUM and CARRY are given below

$$SUM = A \oplus B \oplus C \quad (1)$$

$$SUM = \overline{A \oplus B \oplus C} \quad (2)$$

$$SUM = \left(\overline{\overline{A}B + \overline{B}A}\right)C + \left(\overline{\overline{A}B + A\overline{B}}\right)\overline{C} \quad (3)$$

$$SUM = \left(\overline{\overline{A}\,\overline{B} + AB}\right)\overline{C} + \left(\overline{\overline{A}\,\overline{B} + AB}\right)C \quad (4)$$

$$CARRY = (A \oplus B)C + AB \quad (5)$$

$$CARRY = \left(\overline{\overline{A \oplus B}}\right)C + AB \quad (6)$$

$$CARRY = \overline{\left(\overline{AB} + \overline{(A \oplus B)}\right)\overline{C}} \quad (7)$$

$$CARRY = \overline{\overline{AB} \bullet \overline{AC} \bullet \overline{BC}} \quad (8)$$

$$CARRY = \overline{A \oplus B} \bullet B + A \oplus B \bullet C \quad (9)$$

$$CARRY = \overline{(A \oplus B) \bullet \overline{C} \oplus \overline{AB}} \quad (10)$$

$$CARRY = C \bullet (A \bullet B) + C \bullet (A + B) \quad (11)$$

$$CARRY = AB + AC + BC \quad (12)$$

$$CARRY = \overline{\overline{(A \oplus B)\overline{C}} \bullet \overline{AB}} \quad (13)$$

$$CARRY = AB + \overline{\overline{A \oplus B} \bullet C} \quad (14)$$

$$CARRY = \overline{\overline{AB} \bullet \overline{AC} \bullet \overline{BC}} \quad (15)$$

$$CARRY = \overline{\overline{\overline{A \oplus B} \bullet \overline{AC} \bullet \overline{BC}}} \quad (16)$$

$$CARRY = \overline{\overline{AB} \bullet \overline{\overline{A + B} \bullet C}} \quad (17)$$

$$CARRY = \overline{\overline{A \oplus B \bullet C} \oplus \overline{AB}} \quad (18)$$

$$CARRY = \overline{\overline{A} + \overline{B}} + \overline{\overline{A \oplus B} + \overline{C}} \quad (19)$$

$$CARRY = \overline{\overline{(A \oplus B)C} \oplus \overline{AB}} \quad (20)$$

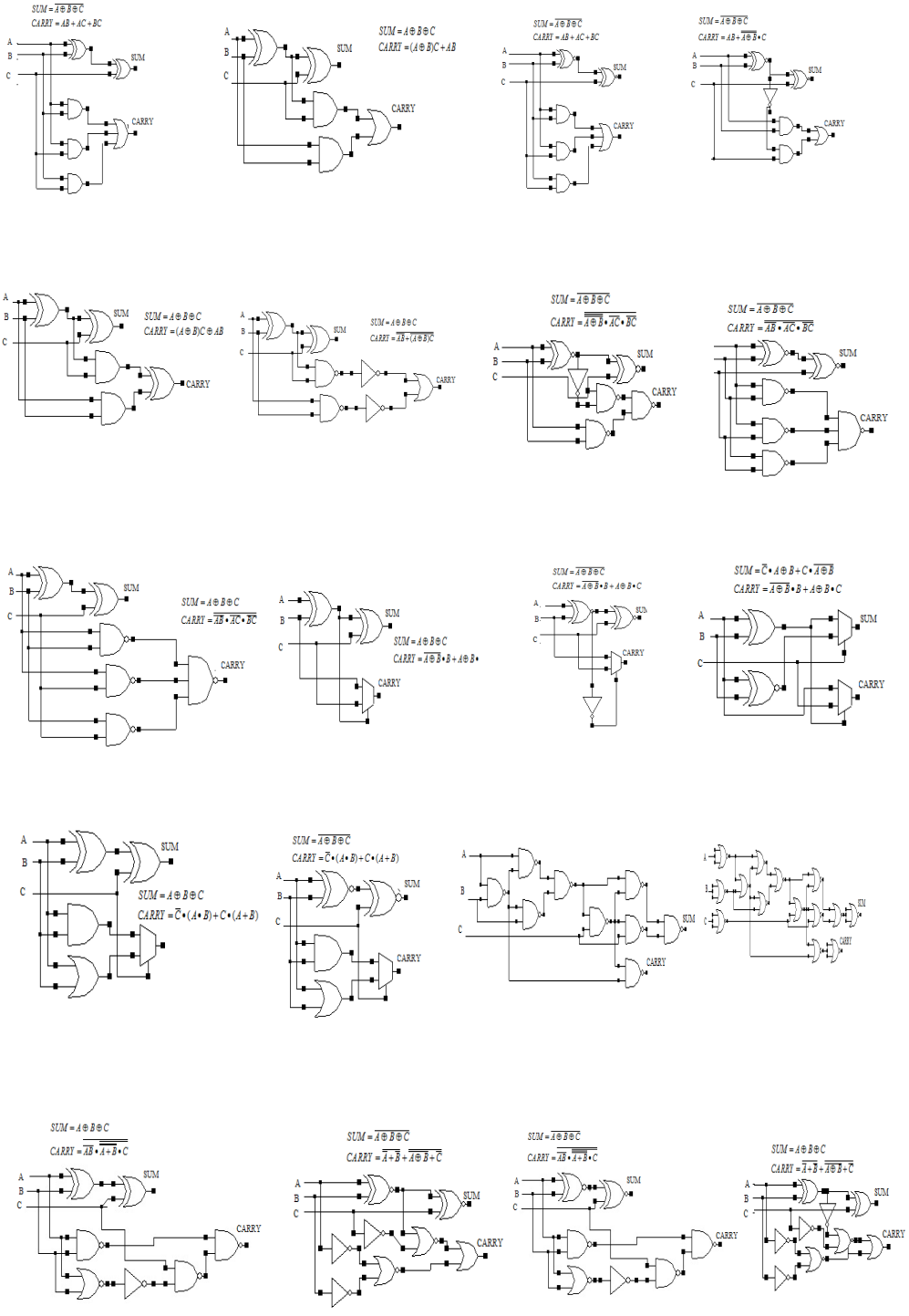


Figure 1 Different structure for realizing Full Adder circuit

In this work 20 different Boolean expressions are formulated (Figure 1 shows different full adder circuit). Using this logical equation it is possible to construct 64 full adder circuits. These adders

are implemented with CMOS logic with technology independent optimization process and its performance are analyzed in terms of transistor count, delay and power dissipation using Tanner EDA with TSMC MOSIS 250nm technology. From this analysis the optimized equation is selected and it is implemented in terms of multiplexers and it is incorporated in selected existing adder topologies like ripple carry adder, carry look-ahead adder, carry skip adder, carry select adder, carry increment adder and carry save adder and its performance is analyzed in terms of area (slices used) and maximum combinational path delay as a function of size.

Mathematically it is also possible to calculate the delay of a circuit by constructing delay models instead of simulation tools using logical effort methods. The logical effort provides a simple method "on the back of an envelope" [13, 14] to choose the best topology (logical constructs) and number of stages of logic for a function. Speed optimization of circuit network can be achieved by the method of logical effort. This method provides how many stages of logic are required for the fastest implementation of any given logic function. The speed of the circuit depends on the capacitive load that the circuit of the logic gate drives and the logic function of the gate. The delay incurred in a logic gate are expressed as sum of two components namely, the parasitic delay p and the effort delay f as follows [13-14].

The delay in single stage network is expressed as

$$d = gh + p \qquad (1)$$

where,
g - Logical effort (the ability of the logic gate's topology to produce output current)
h - Electrical effort (the ratio of output capacitance to input capacitance)
p - Intrinsic delay (delay of the gate due to its own internal capacitance)

Table 1 presents the logical effort of common static CMOS gates assuming the aspect ratio of pull-up and pull-down network to be 2:1 to have equal rise and fall delay. Table 2 presents the parasitic delay of CMOS logic independent of the size of the logic gate and of the load capacitance it drives. The principle contribution to parasitic delay is the capacitance of the source/drain regions of the transistors that drive the gate's output.

Table 1. Logical Effort of CMOS gates

Gate type	Number of Inputs					
	1	2	3	4	5	n
Inverter	1					
NAND		4/3	5/3	6/3	7/3	(n+1)/3
NOR		5/3	7/3	9/3	11/3	(2n+1)/3
XOR/XNOR		4	12	32		

Table 2. Parasitic delay of CMOS gates

Gate type	Number of Inputs					
	1	2	3	4	5	n
Inverter	1					
NAND		2	3	4	5	nPinv
NOR		2	3	4	5	nPinv
XOR/XNOR		4	4	4		4Pinv

An example to calculate the delay of a full adder is shown in Figure (2) using the expression $SUM = A \oplus B \oplus C$ and $CARRY = \overline{\overline{AB} \bullet \overline{AC} \bullet \overline{BC}}$. The circuit is realized as two stage network, stage1 and stage2 respectively. Assume that the input capacitance of 10pf on each input and it will drive the output capacitance with a maximum of 10pf.

Delay calculation for SUM	Delay calculation for CARRY
The Electrical effort H = 10pf/10pf (Output capacitance / Input capacitance) = 1 Path logical effort G along path (N-W) = $\prod g_1 \bullet g_2$ (where g_1 and g_2 are logical effort of XOR gate) =4X4=16 The parasitic delay along path (N-W) P = 4+4=8 (The parasitic delay of XOR gate is 4) The Branching effort B =1 [because all of the fan-out's along the path are one] Number of stages N= 2 stages The path Efforts F=GBH =(16)X1X1=16 The path delay $\hat{D} = N F^{1/N} + P$ = 2X$(16)^{1/2}$+8 =16ps	The Electrical effort H = 10pf/10pf (Output capacitance / Input capacitance) = 1 Path logical effort G along path (N-W) = $\prod g_3 \bullet g_4$ (where g_3 and g_4 are logical effort of NAND gate) =4/3X4/3 = 16/9 The parasitic delay along path (N-W)P =2+2=4 (The parasitic delay of NAND gate is 2) The Branching effort B =1 [because all of the fan-out's along the path are one] Number of stages N= 2 stages The path Efforts F=GBH =(16/9)X1X1=16/9 The path delay $\hat{D} = N F^{1/N} + P$ = 2X$(16/9)^{1/2}$+4 =6.6ps

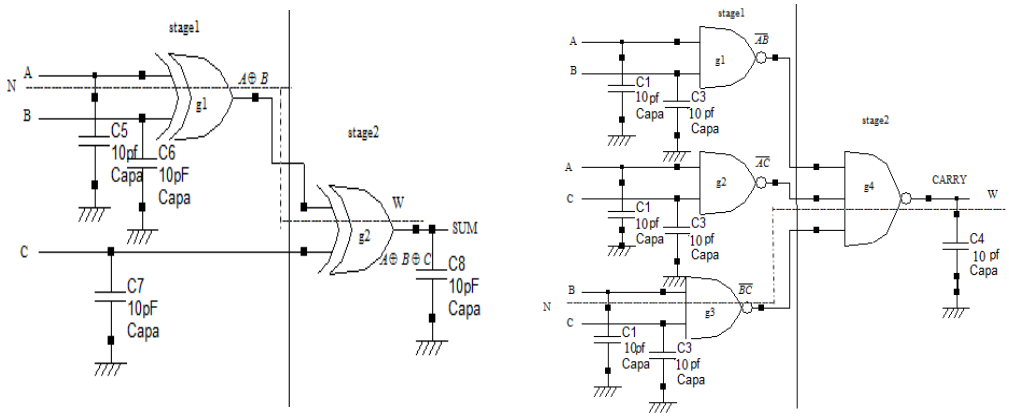


Figure 2. Logical Delay Model for Full Adder Circuit.

So the total delay will be the sum of CARRY and SUM which is equal to 22.6ps. From this observation the delay of the circuit vary with change in the input and output capacitance value.

4. SIMULATION AND PERFORMANCE ANALYSIS OF FULL ADDER

The proposed 20 different Boolean expressions (logic construction) are simulated using Tanner EDA with BSIM3v3 250nm technology with supply voltage ranging from 1V to 2V in steps of 0.2V. All the full adders are simulated with multiple design corners (TT, FF, FS, and SS) to verify that operation across variations in device characteristics and environment. The simulated setup for optimized full adder's (using XOR,MUX) test bed and its gate equivalent along with its input/output waveform is shown in Figure (3). The test bed is supplied with a nominal voltage of 2V in steps of 0.2V and it is invoked with the technology library file Generic 025 and it is specified with TT, FF, FS and SS conditions. The W/L ratios of both nMOS and pMOS transistors are taken as 2.5/0.25µm. To establish an unbiased testing environment, the simulations have been carried out using a comprehensive input signal pattern, which covers every possible transition for a 1- bit full adder.

The frequencies have been chosen in the range from 10 to 200MHz and its input and output capacitances are set to 10pf. The three inputs to the full adder are A, B, C and all the test vectors are generated and have been fed into the adder cell. The cell delay has been measured from the moment the inputs reach 50% of the voltage supply level to the moment the latest of the SUM and CARRY signals reach the same voltage level. All transitions from an input combination to another (total 8 patterns, 000, 001, 010, 011, 100, 101, 110, 111) have been tested, and the delay at each transition has been measured. The average has been reported as the cell delay. The power consumption is also measured for these input patterns and its average power has been reported in Table 3.

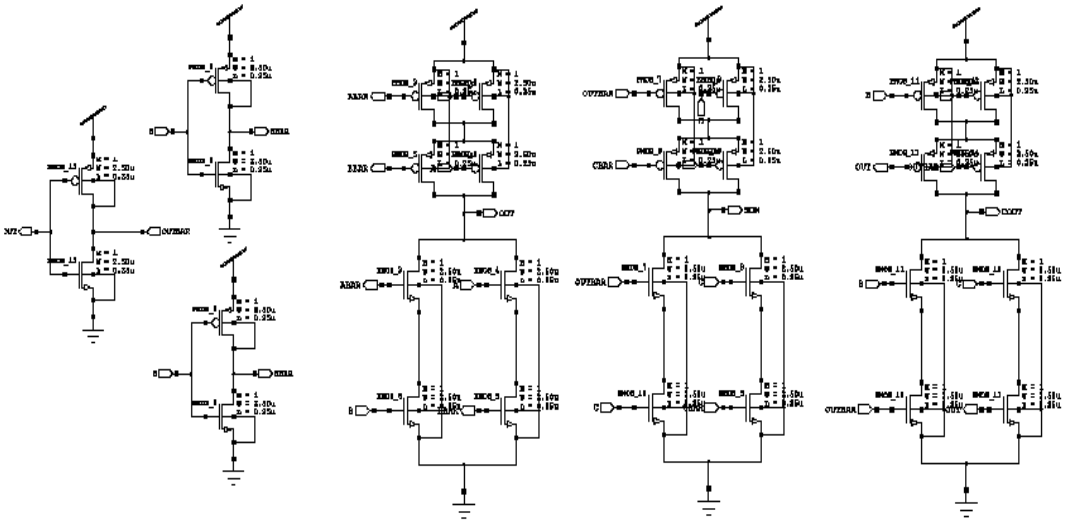


Figure 3. A. Snap Shot of Full Adder with XOR and MUX

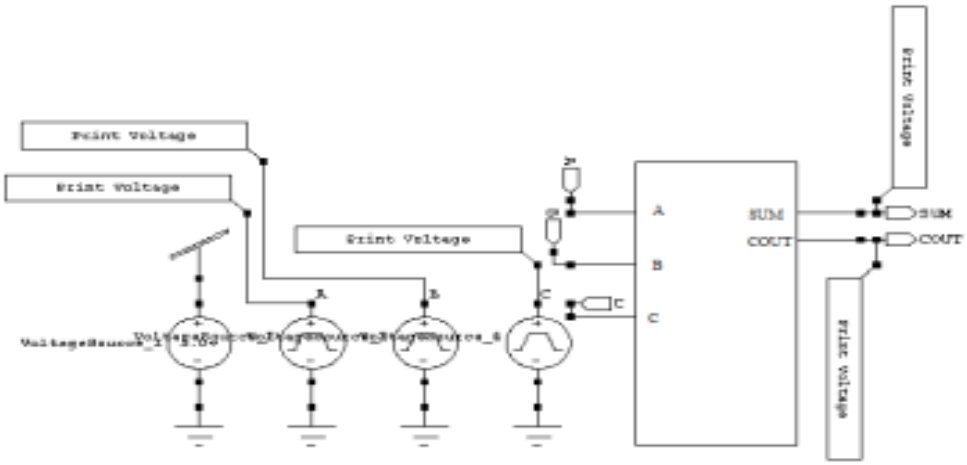


Figure 3.B . Test bed For Full Adder

The simulation results are shown in Table 3. The performance of all the full adders has been analyzed in terms of delay, transistor count and power dissipation. It is observed that adder designed with XOR and MUX has the least delay, transistor count and power dissipation when compared to other combinations of gate. So the adder realized with MUX and XOR is considered

to be the optimized adder in terms of delay, transistor count and power dissipation. The second optimized full adder is realized from XNOR, NOT and MUX.

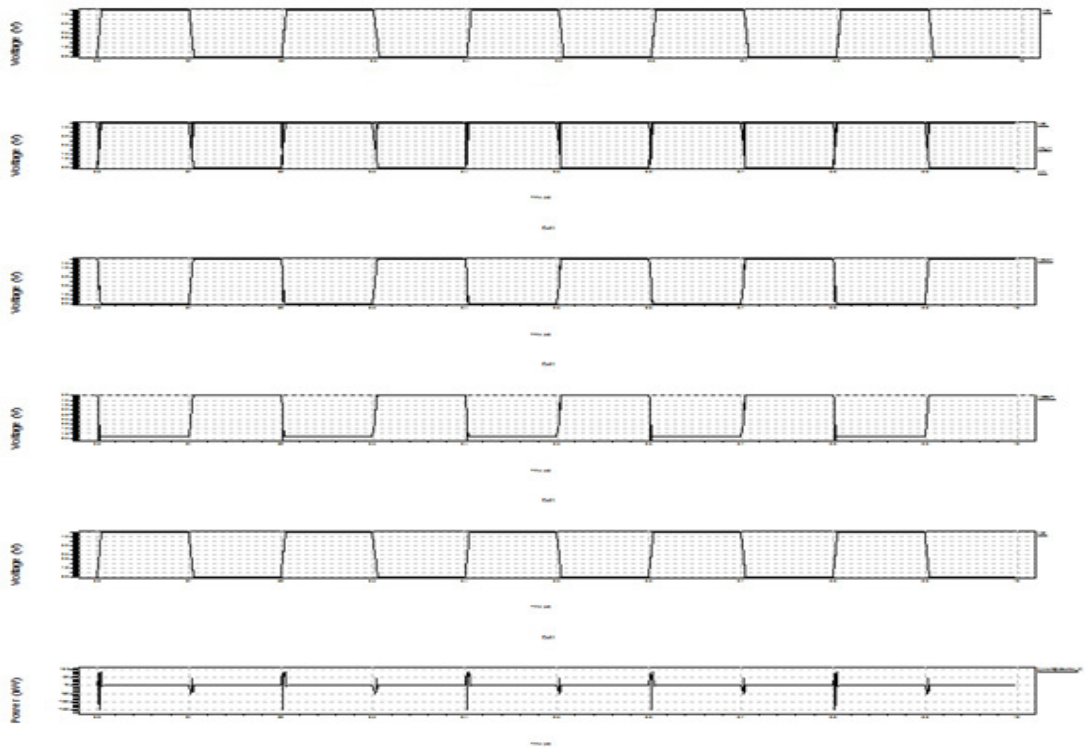


Figure 3. C. Input/ Output Wave of Full Adder

The worst case full adder construction is not using NOR gate which occupies large transistor count, dissipates large power and has longer delay. The other optimized solution for constructing full adders are using NAND gates only, XOR, XNOR,MUX combination and XOR, AND,OR,MUX combination. Figure (4) shows the simulation result of full adders in terms of delay, transistor count (TC) and power dissipation (PWR-DISSP). The power-delay product all the full adders is shown in Figure (5).

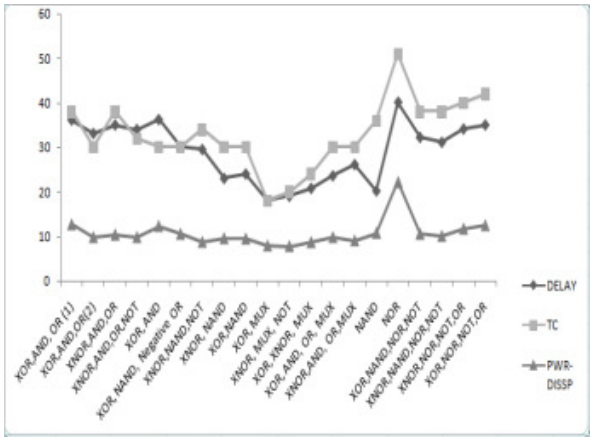


Figure 4. Simulation result of adders in terms delay, area and power

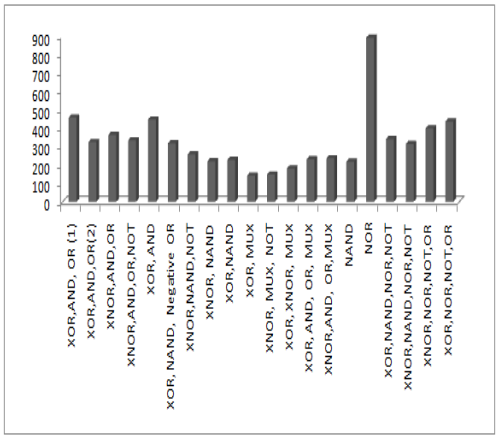


Figure 5. Power delay product

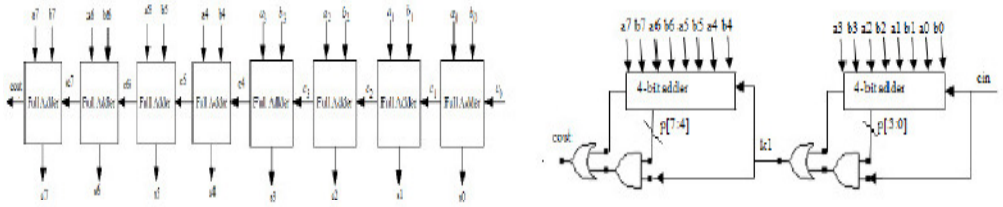
Table 3. Simulated Result for 20 different Full adders

Full adder using	Avg Delay (ps)	Transistor count	Avg Power Dissipation(μW)
XOR,AND, OR (1)	36.1	38	12.693
XOR,AND,OR(2)	33.1	30	9.813
XNOR,AND,OR	35	38	10.371
XNOR,AND,OR,NOT	34	32	9.833
XOR, AND	36.2	30	12.35
XOR, NAND, Negative OR	30.13	30	10.547
XNOR,NAND,NOT	29.5	34	8.752
XNOR, NAND	23.01	30	9.585
XOR,NAND	23.023	30	9.485
XOR, MUX	**18.02**	**18**	**7.704**
XNOR, MUX, NOT	19.01	20	7.769
XOR, XNOR, MUX	20.8	24	8.691
XOR, AND, OR, MUX	23.6	30	9.798
XNOR,AND, OR,MUX	26.12	30	9.058
NAND	20.2	36	10.795
NOR	40.1	51	22.25
XOR,NAND,NOR,NOT	32.2	38	10.583
XNOR,NAND,NOR,NOT	31.1	38	10.1
XNOR,NOR,NOT,OR	34.1	40	11.723
XOR,NOR,NOT,OR	35	42	12.487

5. FPGA IMPLEMENTATION

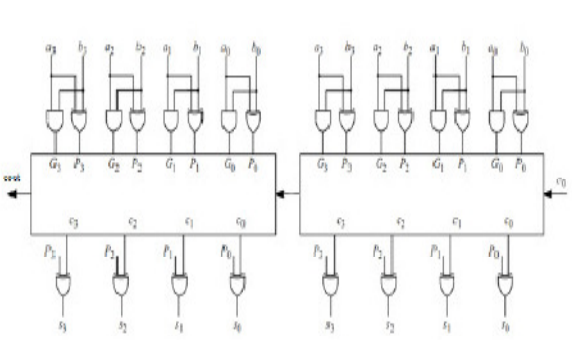
In this work the adder structures used are: Ripple Carry Adder, Carry Look-Ahead Adder, Carry Save Adder, Carry Increment adder, Carry Select Adder, Carry Skip Adder. From section IV it is observed that the optimized equation for implementing 1-bit full adder is using XOR and MUX. So the primitive of this adder cell is implemented with multiplexer and this module is incorporated with existing adder topologies. The target FPGA device chosen for the implementation of these adders was Xilinx ISE 12.1 Spartan3E XC3S500-5FG320. This device was chosen because the Spartan3E families of Field-Programmable Gate Arrays (FPGAs) are specifically designed to meet the needs of high volume, cost-sensitive consumer electronic applications. The Spartan-3E family builds on the success of the earlier Spartan-3 family by increasing the amount of logic per I/O, significantly reducing the cost per logic cell. These Spartan-3E FPGA enhancements, combined with advanced 90 nm process technology, deliver more functionality and bandwidth. Each adder type was implemented with bit sizes of: 8, 16, 32, 64 bits. This variety of sizes will provide with more insight about the performance of each adder in terms of area and delay as a function of size. Structural Gate level modeling using Verilog HDL was used to model each adder. The Xilinx ISE Foundation version 12.1i software was used for synthesis and implementation.

Table 4 contains the results obtained. The adder abbreviations used in the table are: RCA for Ripple Carry Adder, CLA for Carry Look-Ahead Adder, CSA for Carry Save adder, CIA for Carry Increment Adder, CSeA for Carry Select Adder and CSkA for Carry Skip Adder. In the Table, delay is measured in nanoseconds (ns) while area is measured in Slice Look-Up Tables (LUT) units which represent configurable logic units within the FPGA. In the Table E-A represents the adder designed with normal expression for SUM and CARRY, Op-A represents

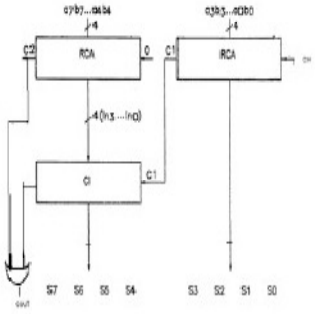


a. 8-bit Ripple Carry Adder (RCA)

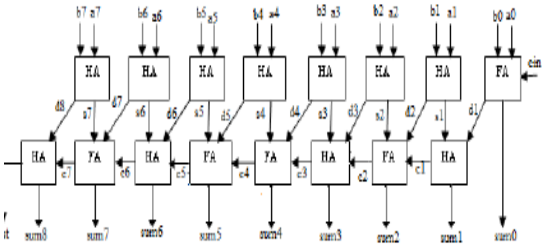
b. 8-bit Carry Skip Adder (CSkA)



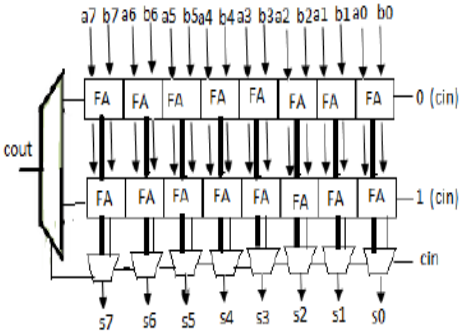
c. 8-bit Carry Look-Ahead Adder (CLA)



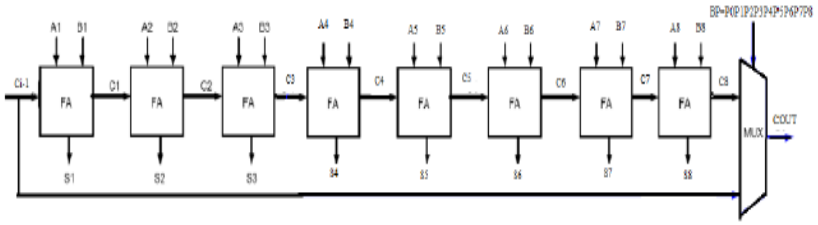
d. Carry Increment Adder (CIA)



b. Carry Save Adder (CSaA)



f. Carry Select Adder (CSelA)



g.Carry Bypass Adder (CByA)

Figure 6 Different Adder topologies

the adder topology implemented with optimized equations that are realized in terms of multiplexers. It is noticed that delay, area and power delay product are less when compared to the normal expression. Figure 6 shows different adder topologies

Table 4 Comparison and Results obtained for different adder topologies

Adders	bit	Delay (ns)		Slice utilized (Area)		AT		Power Dissipation (mw)		PD	
		E-A	Op-A	E-A	Op-A	E-A	Op-A	E-A	Op-A	E-A	Op-A
RCA	8	13.2	**12.93**	9	9	118.83	116.37	80.98	80.1	1069.18	1035.69
	16	21.69	**21.11**	18	18	390.42	379.89	81.1	79.98	1759.06	1687.98
	32	38.67	37.46	37	37	1430.6	1385.87	82.3	78.5	3182.13	2940.3
	64	72.62	70.16	74	74	5373.6	5191.77	85.67	80.2	6221.01	5626.75
CLA	8	13.2	**12.93**	9	9	118.83	116.37	78.91	77.5	1041.85	1002.08
	16	21.69	**21.11**	18	18	390.42	379.89	80.98	78.2	1756.46	1650.41
	32	38.67	**37.46**	37	37	1430.6	1385.87	81.2	79.1	3139.6	2962.77
	64	72.62	**70.16**	74	74	5373.6	5191.77	82.3	79.3	5976.3	5563.61
CSA	8	12.06	**11.1**	14	**13**	168.81	144.3	85.23	80.98	1027.7	898.878
	16	20.02	**19.8**	27	**23**	540.49	455.4	87.1	82.3	1743.57	1629.54
	32	34.94	**32.12**	55	**52**	1921.5	1670.24	88.45	82.3	3090.18	2643.48
	64	56.8	**54.23**	110	**106**	6247.9	5748.38	90.12	84.01	5118.73	4555.86
CIA	8	12.21	**11.9**	12	**11**	146.56	130.9	85.23	80.18	1040.91	954.142
	16	16.57	**14.32**	24	**22**	397.66	315.04	87.2	82.03	1444.82	1174.67
	32	27.45	**25.67**	49	**47**	1345.1	1206.49	88.23	82.03	2422.09	2105.71
	64	47.2	**45.21**	100	**98**	4719.7	4430.58	90.1	84.12	4252.45	3803.07
CSEA	8	12.6	**10.11**	15	**11**	188.94	111.188	78.91	77.5	993.95	783.37
	16	21	**15.75**	32	**30**	672.1	472.56	80.98	78.2	1700.82	1231.81
	32	37.93	**24.36**	67	**61**	2541	1486.2	81.2	79.1	3079.51	1927.19
	64	71.77	**31.41**	135	**125**	9689	8790.12	82.3	79.3	5906.67	2490.65
CSKA	8	12.78	**11.15**	13	13	166.17	144.885	78.91	77.5	1008.63	863.738
	16	23.27	**14.52**	23	**22**	535.28	319.352	80.98	78.2	1884.65	1135.15
	32	40.15	**23.26**	45	**44**	1806.9	1023.22	81.2	79.1	3260.5	1839.47
	64	49.22	**35.73**	79	**78**	3888.5	2787.1	82.3	79.3	4050.97	2833.55

6. SUMMARY

The proposed 20 different Boolean expressions (logic construction) are simulated using Tanner EDA with BSIM3v3 250nm technology with supply voltage ranging from 1V to 2V in steps of 0.2V. It is observed that adder designed with XOR and MUX has the least delay, transistor count and power dissipation when compared to other combinations of gate. So the adder realized with MUX and XOR is considered to be the optimized adder in terms of delay, transistor count and power dissipation. A new low-power, high-speed full adder cell is proposed using XOR and MUX gates. Its performances have been analyzed and reported in section 4. This optimized adder is designed with fully MUX based structure in FPGA using VERILOG HDL and this module is incorporated in the existing adder topologies and its comparison is made. The target FPGA device chosen for the implementation of these adders was Xilinx ISE 12.1 Spartan3E XC3S500-5FG320. The comparison of delay, slice occupied, AT and its power dissipation is depicted in the

Figure (7). From this analysis it is found that for all the adder topologies the delay is less when compare to the existing adder with normal equation ($SUM = A \oplus B \oplus C$ and $CARRY = AB + AC + BC$) and it is also observed that the delay for RCA and CLA are the same and its distribution is shown in the graph (Figure 7a). In case of slice utilized there is no change occurs for RCA and CLA hence its distribution is shown as single red line in the chart (Figure 7b). From AT chart (Figure 7c) it is noticed that the AT value is large for 64 bit carry select adders and adders like ripple carry adder, carry look ahead adder and carry increment adder have less AT Value. From PD distribution (Figure 7d) less power dissipation occurs for carry increment and ripple carry adders, maximum dissipation occurs for carry save and carry skip adders. According to the presented results, the adder topology which has the best compromise between area, delay and power dissipation are carry look-ahead and carry increment adders and they are suitable for high performance and low-power circuits. The fastest adders are carry select and carry save adders with the penalty of area. The simplest adder topologies that are suitable for low power applications are ripple carry adder, carry skip and carry bypass adder with least gate count and maximum delay.

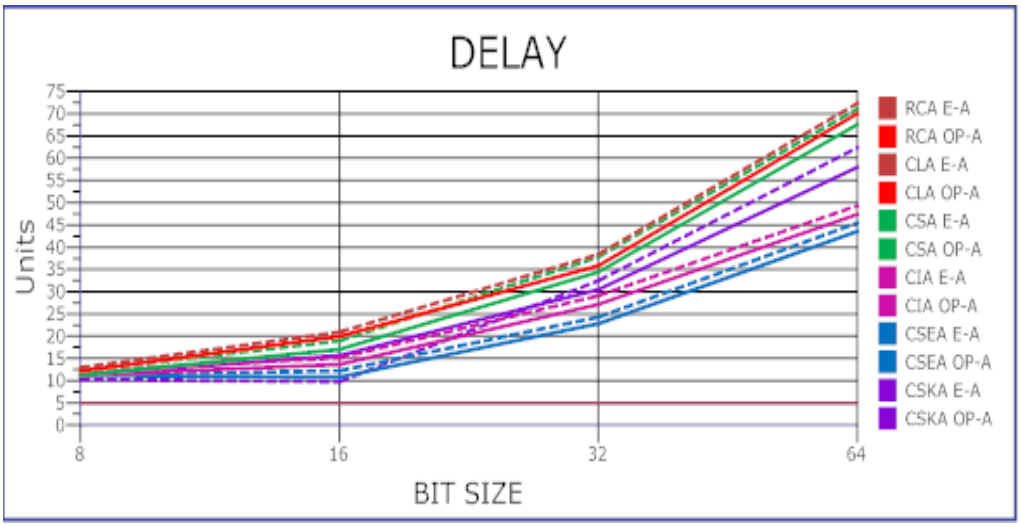


Figure 7. A. Delay Chart

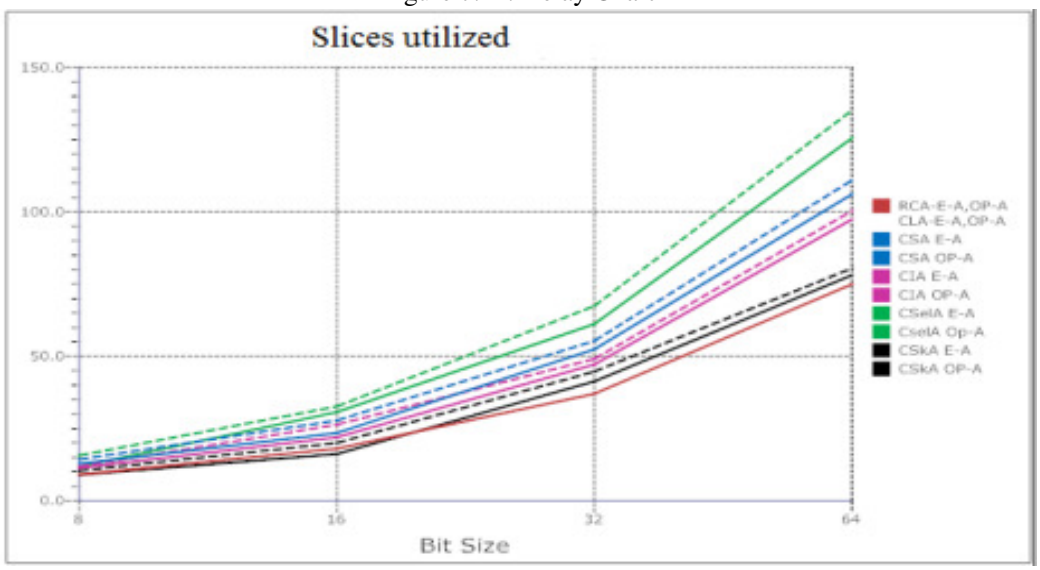


Figure 7. B. Slices utilized Chart

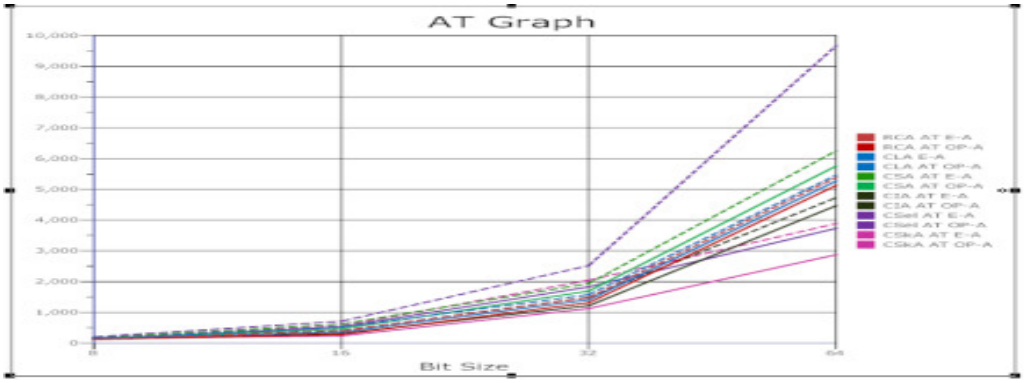


Figure 7.C. AT Value Chart

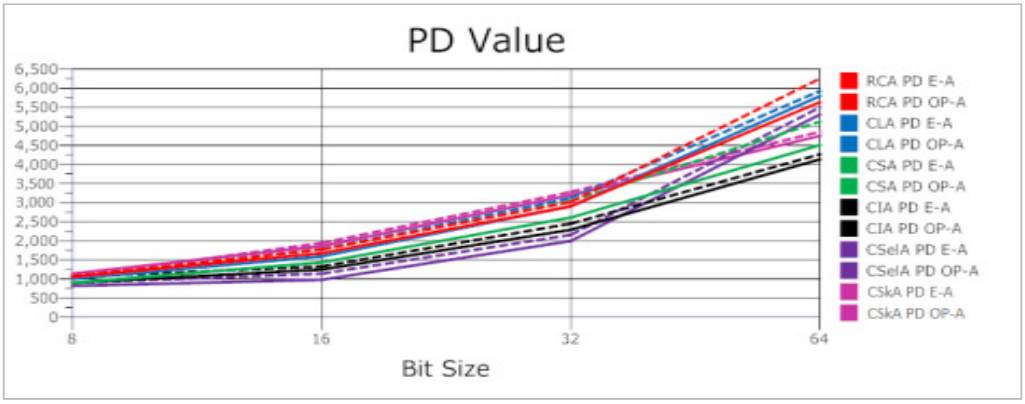


Figure 7. D. PT Value Chart

7. CONCLUSION

An extensive performance analysis of 1-bit full-adder cells has been presented. Technology independent logic optimization is used to design 1-bit full adder with 20 different Boolean expressions and its performance was analyzed in terms of transistor count, delay and power dissipation using Tanner EDA with TSMC MOSIS 250nm technology. From this analysis XOR and MUX based expression provides low transistor count, minimum delay and minimum power dissipation when compared to other logic equations. The second optimized full adder can be realized using XNOR, NOT and MUX. The other optimized solution for constructing full adders are using NAND gates only, XOR, XNOR,MUX combination and XOR, AND,OR,MUX combination. The worst case full adder construction is not using NOR gate which occupies large transistor count, dissipates large power and has longer delay. Logical effort delay model to estimate the parasitic delay is also presented. Using the optimized expression the primitive adder cell is implemented with multiplexer and this module is incorporated with existing adder topologies like ripple carry adder, carry look-ahead adder, carry skip adder, carry select adder, carry increment adder and carry save adder and its performance is analyzed in terms of area (slices used) and maximum combinational path delay as a function of size. The target FPGA device chosen for the implementation of these adders was Xilinx ISE 12.1 Spartan3E XC3S500-5FG320. The comparison and its simulation results have been presented. Based on the comparison it is observed that number of slices occupied, power dissipation and delay are less using the optimized expression. The work presented in this paper gives more insight and deeper understanding of constituting modules of the adder cell to help the designers in making their choices.

REFERENCES

[1] Shih-Chieh Chang, Lukas P.P.P. van Ginneken, "Circuit Optimization by Rewiring", IEEE Transaction on Computers, Vol. 48, No. 9, September 1999.

[2] Oh-Hyeong Kwon, "A Boolean Extraction Technique For Multiple-Level Logic Optimization" IEEE 2003

[3] R.Uma, "4-Bit Fast Adder Design: Topology and Layout with Self-Resetting Logic for Low Power VLSI Circuits", International Journal of Advanced Engineering Sciences and Technology, Vol No. 7, Issue No. 2, 197 – 205, 2011.

[4] Padma Devi, Ashima Girdher and Balwinder Singh, "Improved Carry Select Adder with Reduced Area and Low Power Consumption", International Journal of Computer Application,Vol 3.No.4, June 2010

[5] Shrirang K. Karandikar and Sachin S. Sapatnekar," Fast Comparisons of Circuit Implementations", IEEE Transaction on Very Large Scale Integration (VLSI) Systems, Vol. 13, No. 12, December 2005

[6] Sutherland, B. Sproull, D. Harris, "Logical Effort: Designing Fast CMOS Circuits," Morgan Kaufmann Publisher, c1999.

[7] R.uma, Vidya Vijayan, M. Mohanapriya, Sharon Paul," Area, Delay and Power Comparison of Adder Topologies", International Journal of VLSI and Communication Systems, 2012.

[8] G.Shyam Kishore, "A Novel Full Adder with High Speed Low Area", 2nd National Conference on Information and Communication Technology (NCICT) 2011 Proceedings published in International Journal of Computer Applications® (IJCA).

[9] Mariano Aguirre-Hernandez and Monico Linares-Aranda, "CMOS Full-Adders for Energy-Efficient Arithmetic Applications", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 19, No. 4, April 2011.

[10] Pudi. V, Sridhara., K, "Low Complexity Design of Ripple Carry and Brent Kung Adders in CA",Nanotechnology, IEEE transactions on,Vol.11,Issue.1,pp.105-119,2012.

[11] Shubin.V.V, "Analysis and Comparison of Ripple Carry Full Adders by Speed", Micro/Nano Technologies and Electron Devices (EDM),2010, International Conference and Seminar on, pp.132-135,2010.

[12] Sreehari Veeramachaneni, M.B. Srinivas, "New Improved 1-Bit Full Adder Cells", IEEE, 2008.

[13] Ivan E. Sutherland, Bob F. Sproull, David L. Harris "Logical Effort: Designing Fast CMOS Circuits", Morgan Kaufmann Publishers, Inc. 1998.

[14] R. F. Sproull and I. E. Sutherland. "Theory of Logical Effort: Designing for Speed on the Back of an Envelope". In IEEE Adv. Res. in VLSI, 1–16, 1991.

[15] Jian-Fei Jiang; Zhi-Gang Mao; Wei-Feng He; Qin Wang, "A New Full Adder Design for Tree Structured Arithmetic Circuits", Computer Engineering and Technology(ICCET),2010,2nd International Conference on,Vol.4,pp.V4-246-V4- 249,2010.

[16] Shubhajit Roy Chowdhury, Aritra Banerjee, Aniruddha Roy, Hiranmay Saha, "A high Speed 8 Transistor Full Adder Design using Novel 3 Transistor XOR Gates", International Journal of Electrical and Computer Engineering 3:12 2008.

[15] Romana Yousuf and Najeeb-ud-din, "Synthesis of Carry Select Adder in 65 nm FPGA", IEEE.

[16] D. Patel, P. G. Parate, P. S. Patil, and S. Subbaraman, ―ASIC implementation of 1-bit full adder,‖ in Proc. 1st Int. Conf. Emerging TrendsEng. Technol., Jul. 2008, pp. 463–467.

[17] Y. Sunil Gavaskar Reddy and V.V.G.S.Rajendra Prasad, "Power Comparison of CMOS and Adiabatic Full Adder Circuits", International Journal of VLSI design & Communication Systems (VLSICS) Vol.2, No.3, September 2011

Authors

R.Uma received her B.E. (EEE) degree from Bharathiyar University Coimbatore in the year 1998, Post graduated in M.E (VLSI Design) from Anna University Chennai in the year 2004. Currently she has been working as Assistant Professor in Electronics and Communication Engineering, Rajiv Gandhi College of Engineering and Technology, Puducherry. She authored books on VLSI Design. She has published several papers on national and International journal and conferences. She is the guest faculty for Pondicherry University for M.Tech Electronics. She has received the best teacher award for the year 2006 and 2007. Her research interests are Analog VLSI Design, Low power VLSI Design, Testing of VLSI Circuits, Embedded systems and Image processing. She is a member of ISTE. Perusing her Ph.D. from Pondicherry University in the Department of Computer Science.

Dr. P. Dhavachelvan is working as a Professor in the Department of Computer Science, Pondicherry University, India. He obtained his B.E. in the field of Electrical and Electronics Engineering from University of Madras, India. He pursued his M.E. and Ph.D. in the field of Computer Science and Engineering from Anna University, Chennai, India. He has about 15 years of experience as an academician and his research areas include Software Engineering & Standards and Web Service Computing. In his credit, he has more than 125 research papers published in reputed International and National Journals and Conferences. He also obtained Patents and proposed Standards in the domain of Software Engineering.