International Journal of Scientific & Engineering Research, Volume 4, Issue 5, MAY-2013
ISSN 2229-5518

High-Speed 64-Bit Binary Comparator using Three Different Logic Styles

Anjuli (Student Member IEEE), Satyajit Anand

Abstract--High-speed 64-bit binary comparator using three different logic styles is proposed in this brief. Comparison is most basic arithmetic operation that determines if one number is greater than, equal to, or less than the other number. Comparator is most fundamental component that performs comparison operation. This brief presents comparison of modified and existing 64-bit binary comparator designs concentrating on delay. Means some modifications have been done in existing 64-bit binary comparator design to improve the speed of the circuit. Comparison between modified and existing 64-bit binary comparator designs is calculated by simulation that is performed at 90nm technology in Tanner EDA Tool**.**

Index Terms-- Binary comparator, digital arithmetic, high-speed, logic style, transistor count**.**

1 Introduction

In digital system, comparison of two numbers is an arithmetic operation that determines if one number is greater than, equal to, or less than the other number [1]. So comparator is used for this purpose. Magnitude comparator is a combinational circuit that compares two numbers, A and B, and determines their relative magnitudes (Fig.1). The outcome of comparison is specified by three binary variables that indicate whether A>B, A=B, or A<B. The circuit, for comparing two n-bit numbers, has 2n inputs & 2^{2n} entries in the truth table. For 2-bit numbers, 4-inputs & 16-rows in the truth table, similarly, for 3-bit numbers 6-inputs & 64-rows in the truth table [1].

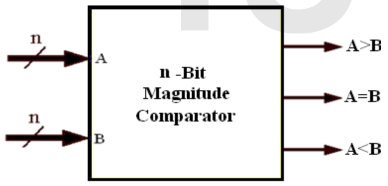

Figure 1. Block Diagram of n-Bit Magnitude Comparator

In recent year, high speed & low power device designs have emerged as principal theme in electronic industry due to increasing demand of portable devices. This tremendous demand is due to popularity of battery operated portable equipments such as personal computing devices, wireless communication, medical applications etc. Demand & popularity of portable electronic devices are driving the designers to strive for higher speed, smaller power consumption and smaller area.

- *Anjuli is currently pursuing masters degree program in VLSI Design in FET-MITS (Deemed University), Lakshmangarh, Sikar, Rajasthan (India),. E-mail: anjuli_222@yahoo.com*
- *Satyajit Anand is currently working as Assistant Professor in E&CE Department, FET-MITS (Deemed University), Lakshmangarh, Sikar, Rajasthan (India). E-mail: satyajitanand45@gmail.com*

The logic style used in logic gates basically influences the speed, size, power dissipation, and the wiring complexity of a circuit [2]. Circuit size depends on the number of transistors and their sizes and on the wiring complexity [3]. The wiring complexity is determined by the number of connections and their lengths. All these characteristics may vary considerably from one logic style to another and thus proper choice of logic style is very important for circuit performance [4].

In order to differentiate both the designs existing and modified, simulations are carried out for delay and power consumption with 1 volt input voltage (and supply voltage), 30^oC temperature and 50MHz frequency at 90nm technology in Tanner EDA Tool.

2 64-Bit Binary Comparator

64-bit binary comparator compares two numbers each having 64 bits (A_{63} to A_0 & B_{63} to B_0). For this arrangement truth table has 128 inputs & 2^{128} entries. By using comparator of minimum number of bits, a comparator of maximum number of bits can be design [5], [6], [7] with the help of tree-based structure logic [8] and also with other useful logic styles.

3 Existing 64-Bit Binary Comparator Design

64-bit comparator in reference [8], [9], [10] represents tree-based structure which is inspired by fact that G (generate) and P (propagate) signal can be defined for binary comparisons, similar to G (generate) and P (propagate) signals for binary additions.
Two number (each having 2-bits: A_1, A_0 & B_1, B_0) comparison can be realized by:

$$B_{Big} = \overline{A_1}\,B_1 + \left(\overline{A_1 \oplus B_1}\right).\left(\overline{A_0}\,B_0\right) \quad (1)$$

$$EQ = \left(\overline{A_1 \oplus B_1}\right).\left(\overline{A_0 \oplus B_0}\right) \quad (2)$$

For A<B, "B_{Big}, EQ" is "1,0". For A=B, "B_{Big}, EQ" is "0,1". Hence, for A>B, "B_{Big}, EQ" is "0,0". Where B_{Big} is defined as output A less than B (A_LT_B). A closer look at equation (1) reveals that it is analogous to the carry signal generated in binary additions. Consider the following carry generation:

$$\mathrm{C_{out}} = \mathrm{AB} + (\mathrm{A} \oplus \mathrm{B}).\mathrm{C_{in}}$$
$$= \mathrm{G} + \mathrm{P}\,.\,\mathrm{C_{in}} \qquad (3)$$

Where A & B are binary inputs C_{in} is carry input, C_{out} is carry output, and G & P are generate & propagate signals, respectively.
After comparing equations (1) & (3):

$$\mathrm{G_1} = \overline{\mathrm{A_1}}\,\mathrm{B_1} \qquad (4)$$
$$\mathrm{EQ_1} = \left(\overline{\mathrm{A_1} \oplus \mathrm{B_1}}\right) \qquad (5)$$
$$\mathrm{C_{in}} = \overline{\mathrm{A_0}}\,\mathrm{B_0} \qquad (6)$$

C_{in} can be considered as G_0. Since for static logic, equation (1) requires tall transistor stack height, hence, an encoding scheme is employed to solve this problem. For this, encoding equation is given as:

$$\mathrm{G_{[i]}} = \overline{\mathrm{A_{[i]}}}\,\mathrm{B_{[i]}} \qquad (7)$$
$$\mathrm{EQ_{[i]}} = \left(\overline{\mathrm{A_{[i]}} \oplus \mathrm{B_{[i]}}}\right) \qquad (8)$$

Where i = 0………..63.
Put these two values from equations (7) & (8) in equations (1) & (2).

$$\mathrm{B_{Big[2j+1:\,2j]}} = \mathrm{G_{[2j+1]}} + \mathrm{EQ_{[2j+1]}}.\mathrm{G_{[2j]}} \qquad (9)$$
$$\mathrm{EQ_{[2j+1:\,2j]}} = \mathrm{EQ_{[2j+1]}}.\mathrm{EQ_{[2j]}} \qquad (10)$$

Where j = 0………..31.
G & P signals can be further combined to form group G & P signals.

$$\mathrm{B_{Big[3:\,0]}} = \overline{\mathrm{A_3}}\,\mathrm{B_3} + \left(\overline{\mathrm{A_3} \oplus \mathrm{B_3}}\right).\left(\overline{\mathrm{A_2}}\,\mathrm{B_2}\right)$$
$$+\left(\overline{\mathrm{A_3} \oplus \mathrm{B_3}}\right).\left(\overline{\mathrm{A_2} \oplus \mathrm{B_2}}\right).\left(\overline{\mathrm{A_1}}\,\mathrm{B_1}\right)$$
$$+\left(\overline{\mathrm{A_3} \oplus \mathrm{B_3}}\right).\left(\overline{\mathrm{A_2} \oplus \mathrm{B_2}}\right).\left(\overline{\mathrm{A_1} \oplus \mathrm{B_1}}\right).\left(\overline{\mathrm{A_0}}\,\mathrm{B_0}\right)$$
$$\mathrm{B_{Big[3:0]}} = \overline{\mathrm{A_3}}\,\mathrm{B_3} + \left(\overline{\mathrm{A_3} \oplus \mathrm{B_3}}\right).$$
$$\left[\overline{\mathrm{A_2}}\,\mathrm{B_2} + \left(\overline{\mathrm{A_2} \oplus \mathrm{B_2}}\right).\left\{\overline{\mathrm{A_1}}\,\mathrm{B_1} + \left(\overline{\mathrm{A_1} \oplus \mathrm{B_1}}\right).\left(\overline{\mathrm{A_0}}\,\mathrm{B_0}\right)\right\}\right]$$
$$\mathrm{B_{Big[3:0]}} = \mathrm{G_3} + \mathrm{EQ_3}.\{\mathrm{G_2} + \mathrm{EQ_2}.(\mathrm{G_1} + \mathrm{EQ_1}.\mathrm{G_0})\}$$

$$\mathrm{B_{Big[3:0]}} = \mathrm{B_{Big[3:2]}} + \mathrm{EQ_{[3:2]}}.\mathrm{B_{Big[1:0]}} \qquad (11)$$
$$\mathrm{EQ_{[3:0]}} = \mathrm{EQ_{[3:2]}}.\mathrm{EQ_{[1:0]}} \qquad (12)$$

Similarly, for 64-bit comparator, B_{Big} & EQ can be computed as:

$$\mathrm{B_{Big[63:0]}} = \mathrm{G_{63}} + \sum_{k=0}^{62}\left(\mathrm{G_k} \cdot \prod_{m=k+1}^{63} \mathrm{EQ_m}\right) \qquad (13)$$

$$\mathrm{EQ_{[63:0]}} = \prod_{m=0}^{63} \mathrm{EQ_m} \qquad (14)$$

Fig. 2 shows 8-bit version of existing tree-based comparator structure and Fig. 3 -Fig. 5 shows corresponding circuit schematics for each logic block of each stage. Pre-encoding circuitry is aimed to minimize the number of transistors. Hence, modified pass transistor logic style is employed to reduce the number of transistors up to 9. In above 8-bit example circuitry, the first stage comparison circuit implements equations (9 & 10) for *j* = 0. . . 3, whereas the second stage generates $B_{Big[3:0]}$, $B_{Big[7:4]}$ and $EQ_{[3:0]}$, $EQ_{[7:4]}$ according to equations (11 & 12). Finally, $B_{Big[7:0]}$ and $EQ_{[7:0]}$ are computed in third stage according to equations (13 & 14).

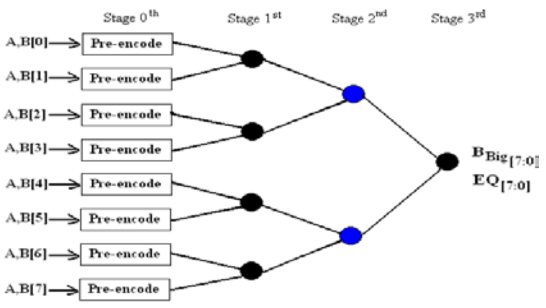

Figure 2. Tree-Diagram of 8-Bit Binary Comparator

Stage 0th is implemented using modified pass transistor logic style giving output in actual form, Stage 1st is implemented using CMOS logic style giving output in inverse form, Stage 2nd is also implemented using CMOS logic style but giving output in actual form.

64-bit comparator is here designed by using 7 stages (from 0th to 6th). In stage 0th, modified pass transistor logic style circuitry (as in Fig. 3) is employed to produce “less than” & “equal to” outputs. Output of stage 0th act as input of stage 1st. In stage 1st, CMOS circuitry (as in Fig. 4) is employed to produce inverse inputs for stage 2nd. In stage 2nd, again CMOS circuitry (as in Fig. 5) is employed to produce actual inputs for stage 3rd. Now, according to tree structure given in Fig. 2, again circuitry of stage 1st is used for stage 3rd. Similarly, for stage 4th, circuitry of stage 2nd is employed. For stage 5th circuitry of stage 1st is employed. For stage 6th circuitry of stage 2nd is employed. Description of this design is given in tabular form in Table 1.

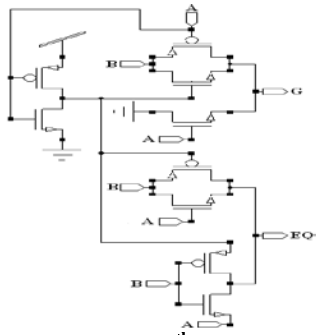

Figure 3. Schematic of Stage 0th of Existing 64-Bit Binary Comparator

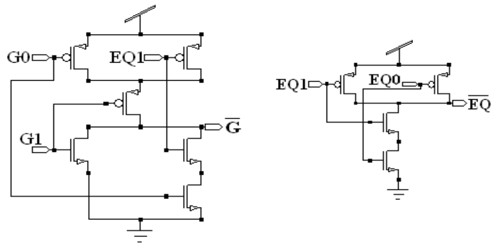

Figure 4. Schematic of Stage 1st of Existing 64-Bit Binary Comparator

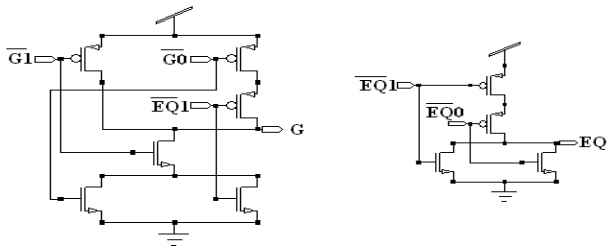

Figure 5. Schematic of Stage 2nd of Existing 64-Bit Binary Comparator

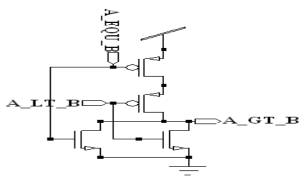

Figure 6. Schematic of NOR gate

According to [8] existing design is having two outputs ("A less than B" & "A equal to B"). This research work also represents here two outputs but they are "A less than B" and "A greater than B". Means "A greater than B" output is here calculated in place of "A equal to B" output. For this arrangement, an extra circuitry of NOR gate (which is shown in Fig. 6) is included at the end of schematic of existing 64-bit binary comparator design. Outputs of "A less than B" & "A equal to B" are given to two inputs of NOR gate that produces "A greater than B" output. Existing design requires 1210 transistor count for 64-bit binary comparator. Accordingly schematic of Existing 64-bit binary comparator is drawn and shown in Fig. 7.

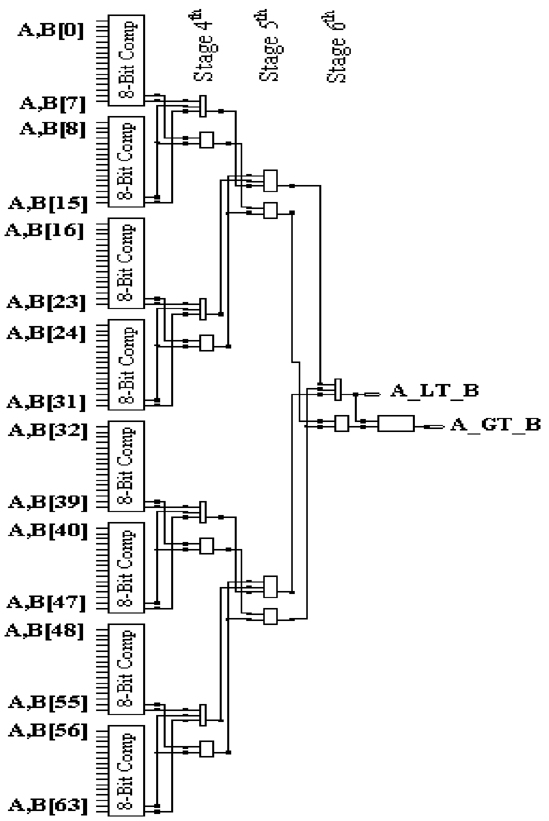

Figure 7. Schematic of Existing 64-Bit Binary Comparator

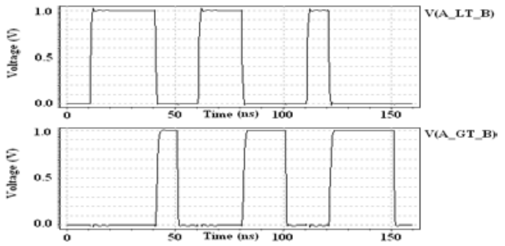

Figure 8. Waveforms of Existing 64-Bit Binary Comparator

According to input bit stream, waveforms of existing 64-bit binary comparator are obtained and shown in Fig. 8. Waveforms show that only one output is high ("1") at a time. When both the outputs "less than" & "greater than" (A_LT_B & A_GT_B) are low ("0"), then waveforms represent that "equal to" output is high (A_EQU_B is "1") at that time. Simulation results for this design are given in Table 4 – Table 6 for conclusion.

4 Modified 64-Bit Binary Comparator Design

Some modifications have been done in existing 64-bit binary comparator design [8] to improve the speed of the circuit. Existing 64-bit binary comparator design [8] follows tree-based structure from 2-bit to 64-bit circuitry. But modified design follows tree-based structure from 2-bit to 8-bit circuitry only. After 8-bit to 64-bit circuitry, modified design follow simple logic structure having two stages (Stage A and Stage B) in place of tree-based structure. In modified design, both the outputs of eight (from 0th to 7th) 8-bit comparators of stage A are given to 8th 8-bit comparator of stage B to produce final outputs ("less than" and "greater than"). A less than B outputs of 0th to 7th 8-bit comparators are given to $A_{0:7}$ inputs of 8th 8-bit comparator. A equal to B outputs of 0th to 7th 8-bit comparators are given to $B_{0:7}$ inputs of 8th 8-bit comparator that produces final outputs.

For this design, In stage A, basic stage 0th is same as existing 64-bit comparator design & implemented using modified pass transistor logic style (Fig. 3) giving output in actual manner. Stage 1st is also implemented using modified pass transistor logic style (MPTL) giving output in actual manner as in Fig. 9. Stage 2nd is same as stage 1st of existing 64-bit comparator design & implemented using CMOS logic style but giving output in inverse manner as in Fig. 10. Description of this design is given in tabular form in Table 2.

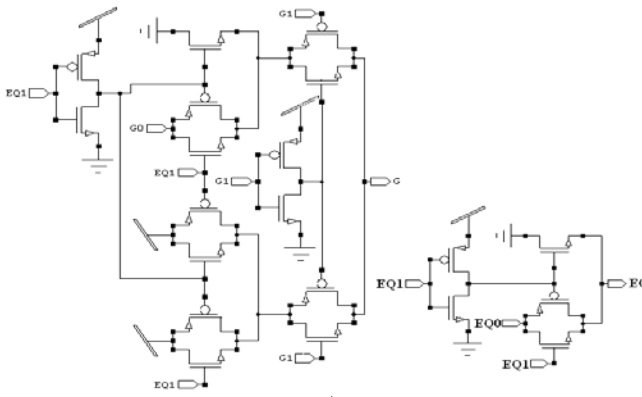

Figure 9. Schematic of Stage 1st of Stage A of Modified 64-Bit Binary Comparator

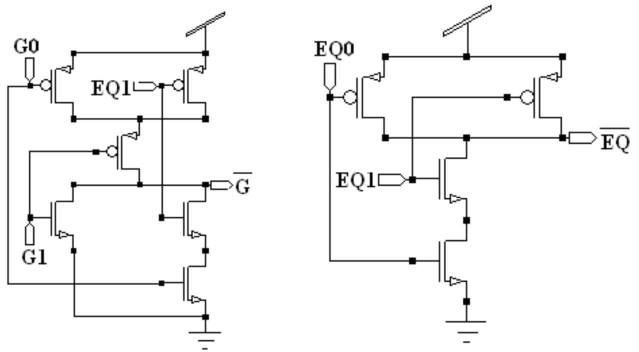

Figure 10. Schematic of Stage 2nd of Stage A of Modified 64-Bit Binary Comparator

In stage B, basic stage 0th is same as existing 64-bit comparator design & implemented using modified pass transistor logic style (Fig. 3) giving output in actual manner. Stage 1st is also same as existing 64-bit comparator design and implemented using CMOS logic style giving output in inverse manner as in Fig. 4. Main idea behind PTL (pass transistor logic) is to use purely NMOS pass transistors network for logic operation [5]. The basic difference of pass-transistor logic style compared to the CMOS logic style is that the source side of the logic transistor networks is connected to some input signals instead of the power lines. In this design style, transistors act as switch to pass logic levels from input to output [4]. But purely NMOS pass transistors network does not provide full output voltage swing. Due to this reason modified pass transistor logic style (MPTL) have been used for stage 0th. MPTL means extra PMOS circuitry is used in pass transistor logic style circuitry to pass logic high ("1") from input to output. Stage 2nd has been implemented using GDI (Gate Diffusion Input) logic style giving output in actual manner as in Fig. 11. The GDI cell contains four terminals – G (the common gate input of the NMOS and PMOS transistors), P (the outer diffusion node of the PMOS transistor), N (the outer diffusion node of the NMOS transistor) and the D node (the common diffusion of both transistors).By using different inputs at P, N and G terminal of GDI cell, the logic gates (AND, OR) can be implemented only with two transistors. Most of these functions require 6–12 transistors in CMOS and other logic styles, but GDI design methodology requires only two transistors per function. GDI enables lower transistor count. Multiple-input gates can be implemented by combining several GDI cells [11], [12]. Description of this design is given in tabular form in Table 3.

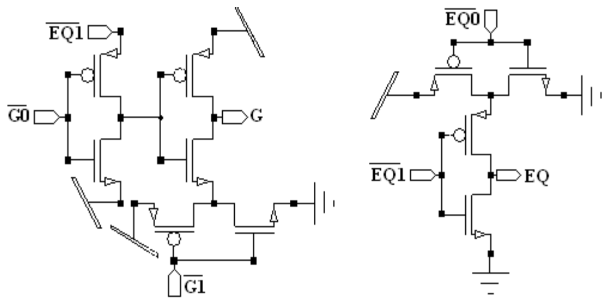

Figure 11. Schematic of Stage 2nd of Stage B of Modified 64-Bit Binary Comparator

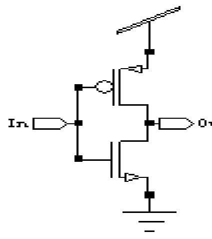

Figure 12. Schematic of Inverter of Modified 64-Bit Binary Comparator

Since output of 8-bit comparators are obtained in inverse form. So, at the end of schematic design of modified 64-bit comparator two inverters (Fig. 12) are required to produce actual form of output waveform. This design requires 1682 transistor count for 64-bit comparator. Schematic (using instances of each section) of modified 64-bit binary comparator design is drawn and shown in Fig. 13.

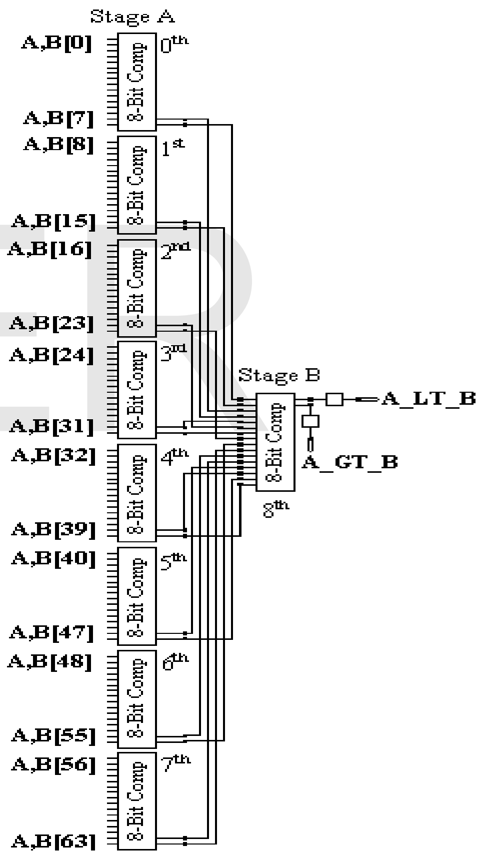

Figure 13. Schematic of Modified 64-Bit Binary Comparator

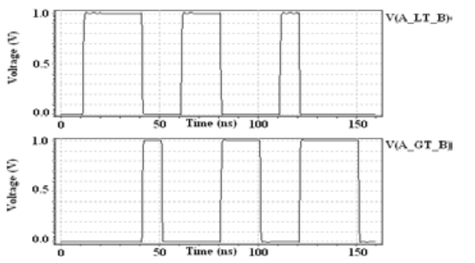

Figure 14. Waveforms of Modified 64-Bit Binary Comparator

According to input bit stream, waveforms of modified 64-bit binary comparator are obtained and shown in Fig. 14.

Input bit stream for modified design is same as in existing design of 64-bit comparator. Output waveforms of modified design produce same position of 1,s and 0,s as in waveforms of existing design for each input bits. Waveforms show that only one output is high ("1") at a time. When both the outputs "less than" & "greater than" (A_LT_B & A_GT_B) are low ("0"), then waveforms represent that "equal to" output is high (A_EQU_B is "1") at that time. Simulation results for modified 64-bit binary comparator design are given in tabular form in Table 4 –Table 6.

5 Simulation And Comparision

After simulation of both the designs final results are obtained for delay and power consumption and are shown in Table 4 – Table 6. Simulations have been carried out at 90nm technology in Tanner EDA Tool.

Table 1
Description of Existing 64-Bit Binary Comparator design

Detail	Stage 0th	Stage 1st	Stage 2nd	Transistor Count
Design	Using MPTL Style	Using CMOS Style	Using CMOS Style	1210
Nature of output	Actual	Inverse	Actual	

Table 2
Description of Stage A of Modified 64-Bit Binary Comparator Design

Detail	Stage 0th	Stage 1st	Stage 2nd	Transistor Count
Design	Same as Existing	Using MPTL Style	Same as Stage 1st of Existing	1536
Nature of output	Actual	Actual	Inverse	

Table 3
Description of Stage B of Modified 64-Bit Binary Comparator Design

Detail	Stage 0th	Stage 1st	Stage 2nd	Transistor Count
Design	Same as Existing	Same as Existing	Using GDI style	146
Nature of output	Actual	Inverse	Actual	

Table 4
Simulation Data with 1volt Input Voltage

Design	Power Consumption (watt)	Delay Time (second)	
		$t_{A\ LT\ B}$	$t_{A\ GT\ B}$
Existing	9.0675e-6	4.4240e-9	1.6028e-8
Modified	1.3271e-5	4.2184e-9	1.5736e-8

Table 5
Simulation Data with 30°C Temperature

Design	Power Consumption (watt)	Delay Time (second)	
		$t_{A\ LT\ B}$	$t_{A\ GT\ B}$
Existing	9.2485e-6	4.4187e-9	1.6033e-8
Modified	1.3491e-5	4.2165e-9	1.5736e-8

Table 6
Simulation Data with 50MHz Frequency

Design	Power Consumption (watt)	Delay Time (second)	
		$t_{A\ LT\ B}$	$t_{A\ GT\ B}$
Existing	9.2765e-6	4.4240e-9	1.6020e-8
Modified	1.3271e-5	4.2184e-9	1.5652e-8

After simulation of both the designs final results are obtained for delay and power consumption with 1 volt input voltage. Delay comparison of modified and existing 64-bit comparator designs is shown in Fig. 15 & Fig. 16. Simulated data for these graphs is given in Table 4.

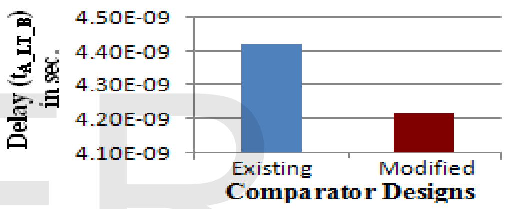

Figure 15. Delay ($t_{A\_LT\_B}$) vs Comparator Designs

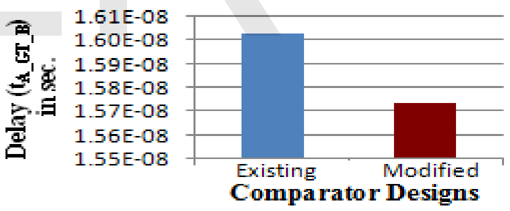

Figure 16. Delay ($t_{A\_GT\_B}$) vs Comparator Designs

The graphs shown in Fig. 15 & Fig. 16 reveal that delay of modified 64-bit comparator design at 1 volt input voltage is remarkably reduced than existing 64-bit comparator design. In Fig.15, delay is reduced 4.6%. In Fig.16, delay is reduced 1.8%.

After simulation of both the designs final results are obtained for delay and power consumption with 30°C temperature. Simulation with temperature has been done at 1 volt input voltage. Delay comparison of modified and existing 64-bit comparator designs is shown in Fig. 17 & Fig. 18. Simulated data for these graphs is given in Table 5.

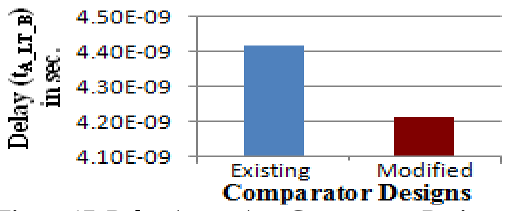

Figure 17. Delay ($t_{A\_LT\_B}$) vs Comparator Designs

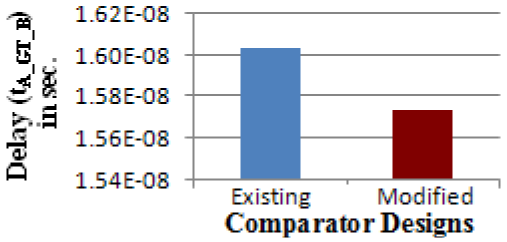

Figure 18. Delay ($t_{A\_GT\_B}$) vs Comparator Designs

The graphs shown in Fig. 17 & Fig. 18 reveal that delay of modified 64-bit comparator design at 30°C temperature is remarkably reduced than existing 64-bit comparator design. In Fig.17, delay is reduced 4.6%. In Fig.18, delay is reduced 1.9%.

After simulation of both the designs final results are obtained for delay and power consumption with 50MHz frequency. Simulation with frequency has been done at 1 volt input voltage. Delay comparison of modified and existing 64-bit comparator designs is shown in Fig. 19 & Fig. 20. Simulated data for these graphs is given in Table 6.

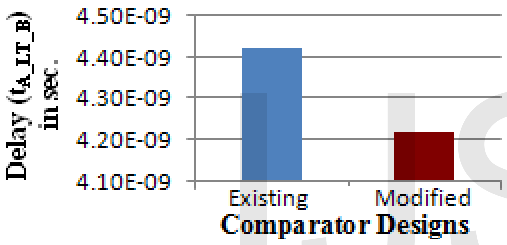

Figure 19. Delay ($t_{A\_LT\_B}$) vs Comparator Designs

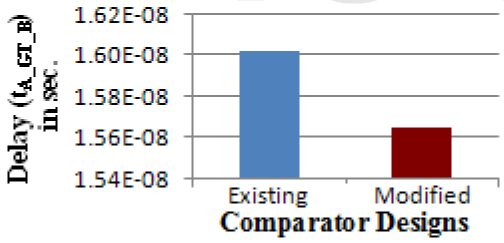

Figure 20. Delay ($t_{A\_GT\_B}$) vs Comparator Designs

The graphs shown in Fig. 19 & Fig. 20 reveal that delay of modified 64-bit comparator design at 50MHz frequency is remarkably reduced than existing 64-bit comparator design. In Fig.19, delay is reduced 4.7%. In Fig.20, delay is reduced 2.3%.

5 Conclusion

In modified design, at 1 volt input voltage delay for output "A less than B" ($t_{A\_LT\_B}$) is reduced 4.6 % and delay for output "A greater than B" ($t_{A\_GT\_B}$) is reduced 1.8 % in comparison to existing design. Similarly, at 30°C temperature delay for output "A less than B" ($t_{A\_LT\_B}$) is reduced 4.6 % and delay for output "A greater than B" ($t_{A\_GT\_B}$) is reduced 1.9 %. And also at 50MHz frequency delay for output "A less than B" ($t_{A\_LT\_B}$) is reduced 4.7 % and delay for output "A greater than B" ($t_{A\_GT\_B}$) is reduced 2.3 % in comparison to existing design. Hence, superiority of modified design is maintained for temperature and frequency also. All of the reduction in delay is obtained after sacrificing power consumption and transistor count. But still modified design gives better result (for delay) than existing design. Therefore, modified 64-bit binary comparator design can be better option for high-speed applications.

Acknowledgment

We are thankful to the Dean, FET and HOD, Department of Electronics & Communication Engineering for providing us necessary permission to carry out this work.

References

[1] M. Morris Mano "Digital Design" *Pearson Education Asia*. 3rd Ed, 2002.

[2] R. Zimmermann and W. Fichtner, "Low Power Logic Styles: CMOS Versus Pass Transistor Logic" *IEEE Journal of Solid State Circuits*, Vol.32, No.7,pp1079-1090,July 1997.

[3] S. Kang and Y. Leblebici "CMOS Digital Integrated Circuit, Analysis and Design" *Tata McGraw-Hill*, 3rd Ed, 2003.

[4] A. Bellaouar and Mohamed I. Elmasry, "Low Power Digital VLSI Design: Circuits and Systems" *Kluwer Academic Publishers*, 2nd Ed, 1995.

[5] C.-H. Huang and J.-S. Wang, "High-performance and power-efficient CMOS comparators," *IEEE J. Solid-State Circuits*, vol. 38, no.2, pp. 254–262, Feb. 2003.

[6] H.-M. Lam and C.-Y. Tsui, "High-performance single clock cycle CMOS comparator," *Electron. Lett.*, vol. 42, no. 2, pp. 75–77, Jan. 2006.

[7] H.-M. Lam and C.-Y. Tsui, "A MUX-based high-performance single cycle CMOS comparator," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 54, no. 7, pp.591-595 July 2007.

[8] Pierce Chuang, David Li, and Manoj Sachdev, Fellow, IEEE "A Low-Power High-Performance Single-Cycle Tree-Based 64-Bit Binary Comparator" *IEEE Transactions on Circuits And Systems—II: Express Briefs*, Vol. 59, No. 2, February 2012.

[9] F. Frustaci, S. Perri, M. Lanuzza, and P. Corsonello, "A new low-power high - speed single – clock -cycle binary comparator," *in Proc. IEEE Int. Symp. Circuits Syst.*, pp. 317–320, 2010.

[10] S. Perri and P. Corsonello, "Fast low-cost implementation of single-clock-cycle binary comparator," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 55, no. 12, pp. 1239–1243, Dec. 2008.

[11] Arkadiy Morgenshtein, Alexander Fish, and Israel A. Wagner, "Gate-Diffusion Input (GDI): A Power-Efficient Method for Digital Combinatorial Circuits" *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*, Vol. 10, No. 5, pp. 566–581, October 2002.

[12] Arkadiy Morgenshtein, Student Member, IEEE, Michael Moreinis, and Ran Ginosar, Member, IEEE, "Asynchronous Gate-Diffusion-Input (GDI) Circuits" *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*, Vol. 12, No. 8, pp. 847–856, August 2004.